Erkki Mäkinen (toim.)

Tietojenkäsittelytieteellisiä tutkielmia
Syksy 2012

INFORMAATIOTIETEIDEN YKSIKKÖ
TAMPEREEN YLIOPISTO

INFORMAATIOTIETEIDEN YKSIKÖN RAPORTTEJA 17/2012

TAMPERE 2012

TAMPEREEN YLIOPISTO
INFORMAATIOTIETEIDEN YKSIKKÖ
INFORMAATIOTIETEIDEN YKSIKÖN RAPORTTEJA 17/2012
JOULUKUU 2012

Erkki Mäkinen (toim.)

Tietojenkäsittelytieteellisiä tutkielmia
Syksy 2012

INFORMAATIOTIETEIDEN YKSIKKÖ
33014 TAMPEREEN YLIOPISTO

ISBN 978-951-44-9031-6

ISSN-L 1799-8158
ISSN 1799-8158

Sisällysluettelo

Tutkimukset aktiivisten vanhusten tietotekniikan käytöstä

Johanna Erkkilä

Tiivistelmä.
Tämä tutkielma tarkastelee aktiivisten vanhusten tietotekniikan käyttöä ja sitä, kuinka aihetta eri tutkimuksissa käsitellään. Tarkoituksena on selvittää, millä käsitteillä aktiivisiin vanhuksiin viitataan, kuinka käsitteiden käyttö toisistaan poikkeaa ja minkälaista teknologiaa heidän kohdallaan tutkitaan.

Avainsanat ja -sanonnat: aktiiviset vanhukset, vanhemmat ihmiset, vanhukset, tietotekniikka, teknologia

CR-luokat: J.7

1. Johdanto

Ikääntyneiden suomalaisten määrä on jatkuvassa kasvussa. Yli 65-vuotiaiden osuuden väestössä arvioidaan nousevan nykyisestä 18 prosentista 26 prosenttiin vuoteen 2030 mennessä [SVT, 2012]. Samankaltainen väestörakenteen muutos on ollut näkyvillä monessa muussakin länsimaassa. Tämä keskimääräinen eliniän pidentyminen ja samanaikainen yhteiskunnallinen kehitys ovat johtaneet uuden väestöryhmän syntymiseen. Aiemmin ihmiset joutuivat usein osallistumaan työntekoon niin pitkään kuin se oli mahdollista. Nykyisin suurin osa eläkkeelle jäävistä on hyvässä kunnossa sekä fyysisesti että psyykkisesti. On alettu puhua kolmannesta iästä, jolla tarkoitetaan työiän ja passiivisen vanhuuden välissä olevaa jaksoa, jolloin ei olla enää työelämässä mutta muuten ollaan aktiivisia.

Samaan aikaan tietotekniikan ja teknologian määrä ja merkitys yhteiskunnassamme on lisääntynyt huomattavasti. Erilaiset laitteet ja koneet ovat tästä näkyvä esimerkki. Laitteiden käytön osaamista edellytetään ihmisten arkipäiväisessä elämässä yhä enemmän, ja monet palvelut ovat siirtymässä enenevissä määrin sähköisiksi. Selviytyäkseen osana yhteiskuntaa ihmisen on kyettävä käyttämään erilaisia teknologisia sovelluksia. Tietotekninen osaaminen nousee näin ollen tärkeään rooliin, eikä sen ulkopuolelle ole varaa jättäytyä, jos haluaa toimia aktiivisena osana yhteiskuntaa.

Tässä työssä keskitytään aktiivisten vanhusten käyttämän tietotekniikan tutkimiseen. Aktiivisilla vanhuksilla viitataan ikääntyneisiin ihmisiin, joiden toimintakyky selvitä arkisten asioiden hoidosta on hyvä. He ovat fyysisesti ja psyykkisesti edelleen kunnossa. Näin ollen he ovat kykeneviä itse käyttämään erilaisia teknisiä laitteita ja sovelluksia.

Passiivisessa vanhuudessa toimintakyky on puolestaan merkittävästi alentunut, eivätkä vanhukset enää kykene tulemaan toimeen omillaan. Passiiviseen vanhuuteen liittyvä teknologia onkin tässä työssä jätetty kokonaan analyysin ulkopuolelle. Tarkastelussa ei siis käsitellä vanhustenhoidossa apuna käytettyä teknologiaa, jonka käyttäjinä usein toimivat muut kuin vanhukset itse. Myöskään puhdasta apuvälineteknologiaa ei ole analyysiin sisällytetty. Toimintakyvyn heikentyminen on hyvin yksilöllistä ja se ilmenee eri ihmisillä eri tavoilla. Selkeää rajanvetoa aktiivisten ja passiivisten vanhusten välille on välillä melko vaikea tehdä. Tässä tutkimuksessa aktiivisiksi vanhuksiksi on määritelty ne henkilöt, jotka pystyvät itse käyttämään teknologiaa.

Tämä tutkielma on kirjallisuuskatsaus. Työssä tarkastelun kohteena on aktiivisista vanhuksista käytetyn terminologian tutkiminen sekä perehtyminen aktiivisten vanhusten käyttämään teknologiaan. Tarkoituksena on selvittää, millaisilla termeillä vanhempaan väestöön tutkimuksissa viitataan ja minkälaisissa yhteyksissä mitäkin termiä käytetään. Toinen painopiste tarkastelussa keskittyy siihen, millaista teknologiaa vanhuksista puhuttaessa nykypäivänä tutkitaan. Lopuksi tehdään ehdotus aihetta käsittelevästä jatkotutkimuksesta sekä yhteenveto.

2. Menetelmä

Tämän kirjallisuuskatsauksen kannalta relevantteja artikkeleita lähdettiin etsimään tietokantahakujen avulla. Tutkielman tarkoituksena on aiempia tutkimuksia analysoimalla tehdä katsaus siihen, minkälaisia käsitteitä aktiivisista vanhuksista tutkimuksissa käytetään ja minkälaista teknologiaa näissä artikkeleissa käsitellään.

Tutkimushakuja tehtiin kahdesta tietokannasta, Springerin elektronisista lehdistä sekä ScienceDirect-artikkeliviitekannasta. Nämä haut ja käytetyt hakusanat koottiin taulukoksi, jossa hakuprosessit ja hyödyllisten artikkelien löytyminen näkyy yksityiskohtaisesti. Tietokantahauissa löytyneet artikkelit käytiin läpi ja niistä löydetyt olennaiset tulokset kerättiin kahteen erilliseen taulukkoon. Toiseen taulukkoon listattiin aktiivisista vanhuksista käytetyt käsitteet ja

toiseen tutkimuksessa esiintynyt teknologia. Seuraavassa kerrotaan, kuinka tutkimukseen päätyneet artikkelit valittiin ja kuinka niitä analysoitiin.

2.1 Artikkelien valinta

Hakujen tekeminen keskitettiin tietokannoissa ensisijaisesti tieteellisiin julkaisuihin. Analysoitavia artikkeleita etsittäessä ja hakuprosessin edistyessä käytettyjen hakusanojen välille kehittyi huomattavia eroja. Osa hakusanoista paljastui toisia hyödyllisemmiksi niiden tuottaessa enemmän halutunlaisia hakutuloksia, kun taas osa termeistä tuotti aiheen kannalta jopa täysin vääränlaisia hakulistauksia. Seuraavassa esitellään tarkemmin käytetyt hakusanat ja niihin liittyvät piirteet niin aktiivisten vanhusten kuin teknologiankin osalta.

Englanninkielisinä hakusanoina toimivat vanhuuteen liittyvät termit ”older people”, ”older population”, ”older adults”, elderly, aged ja ”senior citizens”. Parhaiksi vanhuuteen liittyvistä hakusanoista osoittautuivat ”older adults” (analysoitavista artikkeleista 10 kappaletta, eli noin 29% löytyi tällä hakusanalla), ”older people” (12 kappaletta, 34%) ja ”senior citizens” (7 kappaletta, 20%). Ennakkoon ajatelluista käsitteistä elderly ja aged puolestaan osoittautuivat aiheen kannalta ongelmallisimmiksi. Erityisen toimimattomaksi osoittautui käsite aged, sillä ikääntyneisiin ihmisiin viittaamisen ohella sitä käytetään runsaasti myös muihin ikiin viittaavissa yhteyksissä. Esimerkkinä voidaan mainita school-aged (kouluikäiset) ja middle-aged (keski-ikäiset). Elderly-käsite sinällään tuotti paljon hakuja teknologiaan liittyvien termien kanssa, mutta nämä löytyneet tutkimukset käsittelivät pääosin passiivista vanhuutta ja apuvälineteknologiaa. Lopulliseen analyysiin päätyi vain kaksi artikkelia, joissa tätä käsitettä käytettiin, mikä on vajaa kuusi prosenttia kaikista julkaisuista (taulukko 1).

Tietotekniikkaan liittyviä hakusanoja olivat puolestaan ”information technology”, HCI (human-computer interaction), ”computer use”, television ja ICT (information and communications technology). Näissä hauissa ei ollut nähtävillä yhtä selkeitä eroja eri hakusanojen välillä. Tämä johtunee siitä, että hakukriteerit eivät olleet teknologian suhteen yhtä tiukkoja kuin käyttäjäryhmän. Toisin sanoen tutkimuksen kannalta olennaista on, että käyttäjät ovat aktiivisia vanhuksia, mutta heidän käyttämänsä teknologia saa olla lähes minkälaista tahansa. Teknologiaan viitanneet hakusanat olivat myös melko yleisluonteisia, esimerkkeinä HCI ja ICT, joilla hakutuloksia myös löytyi runsaasti. Eniten tut-

kimuksissa käsitellään tietokoneita ja internetiä, sillä nämä ovat aiheena 12 julkaisussa, mikä vastaa noin 34% analysoiduista tutkimuksista.

ScienceDirect-tietokannassa hakurajaukseksi asetettiin haun kohdistaminen tiivistelmään, otsikkoon ja avainsanoihin (abstract, title, keywords). SpringerLink-sivustolla rajaus oli mahdollista tehdä lähes samaksi, eli haku kohdistettiin tiivistelmään ja otsikkoon (abstract, title). Näissä yllä mainituissa kohdissa julkaisuista tuodaan tarkasti esille se, mitä ne käsittelevät, jonka jälkeen on helpompi nähdä, käsitteleekö tutkimus oman aiheen kannalta olennaisia asioita. Ilman näitä rajauksia hakutulokset olivat epätarkempia ja niihin sisältyi useammin aiheeseen liittymättömiä julkaisuja, kuten väärää käyttäjäryhmää tai tutkimuksen ulkopuolelle rajattua apuvälineteknologiaa käsitteleviä tutkimuksia. Rajaus siis tehtiin, jotta hakuun listautuisi tutkimuksen kannalta mahdollisimman hyödyllisiä artikkeleita.

Kuten tietokantahakutaulukosta näkyy (taulukko 1), hakutuloksista vain pieni osa sisällytettiin lopulliseen analyysiin. Tutkielman rajattu aihealue oli suurin syy karsiutuneille artikkeleille. Vanhuksiin ja teknologiaan liittyvät haut näyttävät viittaavat usein, paitsi aktiivisten vanhusten käyttämään teknologiaan, myös apuvälineteknologiaan tai vanhustenhoidon avuksi kehitettyihin laitteisiin, jolloin varsinaisina käyttäjinä toimivat muut henkilöt kuin vanhukset. Lääketieteeseen ja terveydenhuoltoon liittyvät hakutulokset olivat näin ollen suurin poiskarsittujen artikkelien ryhmä.

Väärän painopisteen lisäksi artikkeleita karsittiin pois useista muistakin syistä. Molemmissa tietokannoissa hakutuloksiin sisältyi myös muulla kielellä kuin englanniksi kirjoitettuja tutkimuksia ja näitä ei lopulliseen analyysiin sisällytetty. Mukaan eivät päässeet myöskään ne artikkelit, joita yliopiston käyttäjäprofiililla ei päässyt tarkemmin lukemaan. Johtuen hakusanojen kierrättämisestä ja erilaisista yhdistelemisestä, välillä hakutuloksiin listautui eri hakukertojen kohdalla samoja artikkeleita. Nämä luonnollisesti sisällytettiin analyysiin vain kertaalleen.

Lopulliseen analyysiin päätyi 35 artikkelia. Näistä 15 löytyi ScienceDirect-hauilla ja 14 SpringerLink-sivustolta (taulukko 1). Molemmissa tietokannoissa käyttäjälle suositellaan hakutulosten selailun yhteydessä muita samankaltaisia artikkeleita. Näiden suositusten kautta analyysiin päätyi mukaan vielä kuusi artikkelia.

2.2 Artikkelien analyysi

Artikkeleissa analyysin kohteena on ollut koko teksti. Analysointia ei ole rajoitettu esimerkiksi pelkkään tiivistelmään tai tutkielman tuloksiin. Kokonaiskuvan saamiseksi artikkelit sisällytettiin tarkasteluun kokonaisuudessaan. ScienceDirect-tietokannasta on haettavissa suurempi määrä artikkeleita, yli 11 miljoonaa, verrattuna SpringerLinkin vastaavaan reiluun neljään miljoonaan artikkeliin. Käytettyjen tietokantojen välillä ei kuitenkaan artikkelien löytymisen kannalta ollut suurta eroa, sillä molemmista päätyi lopulliseen analyysiin lähes yhtä monta artikkelia. Eroja ei ollut myöskään artikkelien laadussa.

Käsitteiden määrittelyä haettaessa tarkastelu usein keskittyi artikkelien alkuosaan, sillä aktiivisista vanhuksista käytettävät nimitykset on useimmiten pyritty tuomaan esille jo tutkimusten alussa. Samassa yhteydessä nämä käytetyt termit yleensä myös määritellään, useimmiten jonkin ikärajan mukaan. Myös tutkittu teknologia tuodaan useimmiten esille jo johdannossa. Useinkaan teknologiaa ei sen tarkemmin jouduta määrittelemään, sillä jo pelkkä käsite antaa yleensä riittävän selkeän kuvan tutkitusta tekniikasta. Esimerkiksi tietokoneiden käyttöä tutkittaessa ei tämänkaltaisissa tutkimuksissa yleensä keskitytä vain tietyn merkkisiin tai tietyn ikäisiin koneisiin. Teknologian käyttöä ja siihen liittyviä piirteitä käsitellään kuitenkin useimmiten vasta tutkimusten varsinaisessa käsittelyosassa, jolloin myös näiden osuuksien tarkastelu muodostui olennaiseksi osaksi tämän tutkielman kannalta. Näistä löytyneistä tuloksista muodostettiin kaksi taulukkoa, joista toisessa käsitellään aktiivisista vanhuksista käytettyjä nimityksiä ja niiden määrittelyjä ja toisessa tutkittua teknologiaa.

On kuitenkin hyvä huomata, että artikkelit ovat aiheen tarkastelun kannalta keskenään hieman erityyppisiä. Toiset tutkimukset käsittelevät hyvinkin tarkasti vanhusten käyttämää teknologiaa, kun toiset tutkimukset vain sivuavat aihetta asettaen varsinaisen painopisteen toisaalle, esimerkiksi vanhusten hyvinvointiin. Näin ollen osaa artikkeleista luettiin tarkemmin kuin toisia. Eri tutkimuksissa käytetyt menetelmätkin poikkeavat toisistaan varsin huomattavasti, sillä mukaan valikoitui niin kenttätutkimuksia kuin esimerkiksi kirjallisuuskatsauksia. Artikkeleita ei näin ollen ole voinut täysin verrata keskenään, mutta niistä on ollut mahdollista poimia tietyt, tähän tutkimukseen liittyvät teemat omiin taulukoihinsa.

Tietokanta	Hakusanat	Tulos	Mukaan
ScienceDirect	older people AND technology	216	
	older people AND technology	205	
	(”older people” OR ”older population”) AND (technology OR HCI)	66	10
	elderly AND (HCI OR technology OR "computer use" OR ”information technology”)	374	
	elderly AND technology	367	
	elderly AND ”computer use”	4	
	elderly AND HCI	6	
	elderly AND ”information technology”	17	2
	aged AND (”information technology” OR HCI OR computer)	958	
	”senior AND (”information technology” OR HCI OR ”computer use”)	148	
	”senior citizen”AND ”computer use”	0	
	senior citizens” AND (”information technology” OR HCI OR computer)	5	3
SpringerLink	”older people” AND ”information technology”	5	3
	”older adults” AND ”information technology”	4	
	elderly AND "information technology”	17	3
	"older person" OR elderly AND HCI	6	1
	"senior citizens" AND (computer OR HCI)	7	1
	"aged people" AND (computer OR HCI OR ICT)	1	1
	"older adults" AND (tv OR television OR radio)	15	5
Yhteensä (montako löytyi, montako mukaan analyysiin)		2468	29

Taulukko1: Hakutulokset

3. Aktiivisiin vanhuksiin viittaaminen tutkimuksissa

Aktiivisiin vanhuksiin voidaan viitata useilla eri käsitteillä. Nämä käytettävät nimitykset eroavat paikoitellen hieman toisistaan, eikä mitään yleistä ohjesääntöä eri termien käyttöön ole olemassa. Kaskiharju [2004] on esittänyt, että vanhoista ihmisistä käytetty kieli on tilannekohtaista ja kontekstisidonnaista. Eri käsitteisiin liittyy usein erilaisia mielikuvia ja nämä voivat ajan myötä myös muuttua.

Tämä käsitteiden selkeä määrittelemättömyys käy hyvin ilmi aihepiirin tutkimuksia luettaessa. Samalla termillä voidaan eri tutkimuksissa viitata eri-ikäisiin ja toimintakyvyltään hyvinkin eritasoisiin henkilöihin. Tutkimukset voidaankin mielestäni karkeasti jaotella sen mukaan, määritelläänkö käytetty käsite tarkasti esimerkiksi jonkin ikärajan tai elämänvaiheen mukaan, vai käytetäänkö sitä epämääräisemmin vain antamaan osviittaa henkilön iästä.

Tässä tutkielmassa analysoitiin sekä artikkeleita, jotka loivat selkeät ikärajat käyttämälleen nimitykselle että artikkeleita, joissa käsitettä ei sen tarkemmin määritelty. Jopa 20% analysoiduista artikkeleista jättää kokonaan määrittelemättä käyttämänsä käsitteen (taulukko 2). Tällöin tutkielmissa esiintyneiden henkilöiden ikä jää sen varaan, millainen mielikuva lukijalle oman aiemman elämänkokemuksen perusteella on kyseisestä käsitteestä syntynyt. Osassa artikkeleita nimitykset kyllä määritellään, mutta tätä ei tehdä välttämättä kovin tarkasti vaan paremminkin suunta-antavasti. Tällöin saatetaan esimerkiksi puhua yli 60-vuotiaista vanhemmista ihmisistä, kuten Blythe ja muut [2005] ovat esittäneet. Loput julkaisuista antoivat käyttämälleen käsitteelle usein hyvinkin tarkan määritelmän, kuten Sayago ja Blat [2010] viitatessaan 65–80-vuotiaisiin henkilöihin.

Näiden yllä mainittujen toimintatapojen poikkeavuuksien ja käsitteisiin liittyvien tulkintaerojen vuoksi on mahdotonta tehdä tarkkaa listaa tietyistä nimityksistä ja niihin liittyvistä ikärajoista tai muista mielleyhtymistä. Jonkinlaisia karkeita yleistyksiä on kuitenkin mahdollista tehdä perustaen havainnot eri tutkimusten ja artikkelien yhteyksissä oleviin määrittelyihin. Näitä päätelmiä ei kuitenkaan voi pitää lopullisena totuutena, sillä kuten Kaskiharjukin [2004] toteaa, käytetyt sanat ovat sidottuja aikaansa ja asiayhteyteensä.

Artikkeleissa löytyneistä käsitteet ja niiden määritelmät on koottu taulukkoon 2, jossa ne on lueteltu aakkosjärjestyksessä. Taulukon 2 kolmea käytetyintä käsitteitä ja niiden välisiä eroavaisuuksia tarkastellaan yksityiskohtaisemmin. Nämä kolme tarkempaan selitykseen valittua nimitystä kattavat yli 90% kaikista analysoitavina olleista artikkeleista. Lopussa tarkastellaan myös muita esiin tulleita tapoja puhua aktiivisista vanhuksista.

Authors	Year	Used terminology	Definiton
Kang, Kim, Kim	2009	Aged people	No definition
Emiliani	2002	Elderly people	No definition
Pieper, Hermsdorf	1997	Elderly person	≥ 60
Ushida *et al.*	2001	Elderly women	70-89 years
Dickinson, Gregor	2006	Older adults	> 50 years
Eronen	2006	Older adults	≥ 60
Fukuda	2011	Older adults	73.5 years (the average age of 8 people)
Lee, King	2003	Older adults	> 50
Rice, Alm	2007	Older adults	No definition
Rice, Carmichael	2011	Older adults	No definition
Slegers, van Boxtel, Jolles	2012	Older adults	> 50
Von Bruhn Hinné, Keates	2011	Older adults	65-85 years
LeRouge *et al.*	2011	Older adults, elderly patients	Not exact, refers to people aged 65 and older
Kurniawan *et al.*	2006	Older adults, older people	Not exact, refers to people aged 60 to 64
Blythe, Monk, Doughty	2005	Older people	> 60
Dickinson *et al*	2005	Older people	> 60
Eisma *et al.*	2004	Older people	≥ 60
Kurniawan	2008	Older people	≥ 60
Sayago, Blat	2010	Older people	65-80 years
Sayago, Sloan, Blat	2011	Older people	58-77 years
Song, Lee	2008	Older people	No definition
Toepoel	2012	Older people	≥ 55
Zajicek	2006	Older people	No definition
Heart, Kalderon	2011	Older people (also refers to youngest old and oldest old)	Older people (≥65), Youngest old (60-69), Oldest old (≥ 80)
Goodman, Lundell	2005	Older people, older population	> 60
Goodman-Deane, Keith, Whitney	2009	Older population	Not exact but the article refers to people ≥ 65
Chong, Theng	2004	Senior citizen	≥ 50
Escuder-Mollon	2012	Senior citizens	> 55 or retired
Niehaves, Plattfault	2010	Senior citizen	≥ 50
Nora *et al.*	2011	Senior citizens	> 55
Tielen	1998	Senior citizen	No definition
Peacock, Künemund	2007	Senior citizen	≥ 55
Esteller-Curto *et al.*	2012	Senior citizens, senior learners	> 55
Godfrey, Johnson	2009	Younger old, older old	Younger old (60-71) Older old (≥ 71)
Salovaara *et al.*	2010	55-65 years old (late middleagers)	Refers to age range (55 to 65) and avoids using any specific term

Taulukko 2: Nimitykset

3.1 Elderly - vanhus

Ensimmäisenä käsitellään englanninkielistä termiä ”elderly”, joka tässä työssä käännetään vanhukseksi. Tämän työn tarkastelun kohteena ovat aktiiviset vanhukset, eli ne ikäihmiset, jotka tulevat toimeen itsenäisesti. Tutkimuskirjallisuudessa sanalla vanhus tunnuttaan hyvin usein viittaavan henkilöihin, joiden toimintakyky on jo huomattavasti heikentynyt ja jotka usein tarvitsevat apua tullakseen toimeen. Käsitteeseen läheisesti yhdistetty termi näyttää oleva ”elderly care” eli vanhustyö tai vanhustenhoito. Analysoitaviksi kelpuutetuista artikkeleista vain kolmessa, eli vajaassa 9%, käytettiin elderly-käsitettä.

Lähes poikkeuksetta tällä hakusanalla löytyneet tutkimukset tuntuvat keskittyvä passiivisille vanhuksille tarkoitetun apuvälineteknologian sekä terveydenhuoltoon liittyvien laitteiden tutkimiseen ja soveltamiseen. Näin ollen aktiivista vanhuutta käsiteltäessä termiä ei juuri näytetä käyttävän. Pieper ja muut [1997] määrittelevät elderly-termin kattamaan henkilöt 60-vuotiaista ylöspäin. Toisessa tutkimuksessa ikähaarukaksi mainitaan puolestaan 70–89-vuotiaat [Ushida *et al.*, 2001]. Näitä ikärajauksia voidaan pitää hieman korkeampina kuin monen muun käsitteen yhteydessä mainittuja ikämääritelmiä (taulukko 2).

Kansalaisilta kysyttäessä aiemmin mainittua 60 vuoden ikärajaa pidetään kuitenkin liian matalana, sillä heidän ikäisiään ei vielä mielletä vanhuksiksi. Yli 80-vuotiaita ihmisiä käsitteen katsotaan sen sijaan kuvaavan jo paremmin [STM, 1999]. Tämä näyttää tukevan ajatusta siitä, että vanhus sanalla viitataan yleisesti huomattavasti ikääntyneempään ja toimintakyvyltään usein rajoittuneempaan väestöön.

3.2 Older adults, older people - vanhemmat ihmiset

Varsin vakiintunut tapa puhua ikääntyneemmästä väestöstä on viitata heihin käsitteillä ”older adults” ja ”older people”. Suomenkielisiä vastineita ei ole aivan yhtä helppo määrittää kuin edellisessä tapauksessa, mutta tässä yhteydessä käsitteet käännetään vanhemmiksi ihmisiksi. Analysoiduista artikkeleista peräti 22:ssa käytettään näitä nimityksiä (taulukko 2). Se vastaa noin 63% kaikista tarkasteluun otetuista julkaisuista. Nämä käsitteet näyttävät siis olevat hyvin käytettyjä tässä aiheyhteydessä.

Näidenkään kohdalla ei kuitenkaan voida puhua mistään kansainvälisesti sovituista ikärajauksista ja kyseisiin termeihin viitataankin vaihtelevasti aina 50-vuotiaista 80-vuotiaisiin ja siitä ylöspäin. Näitä nimityksiä käyttävistä 22 ar-

tikkelista 18 on määritellyt käsitteen ikärajauksella. Näiden ikärajauksien keskiarvoksi tuli noin 62 vuotta, eli keskimäärin näillä termeillä tunnutaan viittaavan hieman yli 60-vuotiaisiin ihmisiin.

Vanhus-käsitteeseen verrattuna huomionarvoista kuitenkin on, että tällä nimityksellä hakutuloksiin ei listautunut yhtä paljon terveydenhuoltoon ja passiiviseen vanhuuteen liittyviä tutkimuksia. Tämä näkyy muun muassa analysoitaviksi päätyneiden julkaisujen suurempana määränä (taulukko 1). Vanhemmat ihmiset -käsitteellä tunnutaan siis usein viittaavat toimintakyvyltään paremmassa kunnossa oleviin ihmisiin kuin vanhus-termillä.

3.3 Senior citizens - seniorit

Käsite senior tarkoittaa suoraan suomennettuna vanhempaa tai vanhinta. Termi voitaisiin suomentaa vapaammin senioriksi tai seniorikansalaiseksi. Pelkkä seniori sanan käyttäminen hakuterminä ei osoittautunut kovin toimivaksi, sillä varsinkin englannin kielessä sana ”senior” viittaa iältään vanhempien lisäksi myös esimerkiksi työelämän arvoasteikossa korkeammalla oleviin, jolloin hakutuloksiin listautui paljon muun muassa hallintoon ja johtamiseen liittyviä artikkeleita. Seniori-käsitteeseen näytetään usein siis liittävän varsin arvostava ajatus kokeneesta henkilöstä.

Senioria kuitenkin käytetään tutkimuksissa jonkin verran myös pelkkään ikään viittaavana terminä. Tämä käy esille hyvin silloin, kun hakutermiä rajataan tarkempaan määritelmään, ”senior citizens”. Termi ei tuntunut olevan käytetyimpien joukossa, sillä hakutulosten määrä jäi huomattavasti vähäisemmäksi kuin kahden aiemmin esitellyn käsitteen kohdalla (taulukko 1). Tosin haussa esiin tulleet artikkelit olivat keskimäärin aiheen kannalta hyvin relevantteja ja suurin osa niistä sisällytettiin mukaan analyysiin. Tutkituista julkaisuista 20% käyttää käsitettä seniori.

Yksi selkeä piirre seniori-käsitteen käyttöön kuitenkin liittyy. Toisin kuin esimerkiksi vanhus-termin kohdalla, seniorissa ikäraja tuntuu ulottuvan usein vielä työelämässä oleviin asti. Useissa artikkeleissa senioreina pidetään pääsääntöisesti 55-vuotiaita ja vanhempia, mutta joissain tutkimuksissa ikäraja on laskettu jopa 50-vuotiaista alkavaksi [Niehaves and Plattfault, 2010]. Tämä ikämääritelmä näyttää kuitenkin olevan melko yhtenäinen, sillä kaikki analyysiin päätyneet artikkelit määrittelivät ”senior citizen” -käsitteen alarajaksi joko 50- tai 55-vuotiaat (taulukko 2).

3.4 Muut viittaustekniikat

Yllä mainittujen käsitteiden lisäksi analysoitavista artikkeleista löytyy myös muunlaisia tapoja viitata aktiivisiin vanhuksiin. Nämä tavat muodostavat kuitenkin selkeän vähemmistön artikkelien joukossa, sillä kolme yleisintä viittaustapaa kattavat yli 90% analysoiduista artikkeleista (taulukko 2).

Muista julkaisuissa esille tulleista käsitteistä mainittakoon ensimmäisenä ”aged person”, joka käännetään tässä yhteydessä ikääntyneeksi ihmiseksi. Termi ei osoittautunut tämänkaltaisessa tutkimuksessa kovinkaan käytetyksi ja suurin syy tähän oli varmasti sen englanninkielisen aged-sanan viittaaminen myös muunikäisiin ihmisiin (esimerkiksi jo aiemmin mainittu school-aged, kouluikäinen). Ainoastaan Kang ja muut [2009] käyttävät kyseistä termiä aktiivisiin vanhuksiin viitatessaan, mutta hekään eivät anna käsitteelle sen tarkempaa määrittelyä.

Osassa artikkeleista on haluttu kiinnittää tarkempaa huomiota siihen, että yksittäinen termi saattaa kattaa hyvinkin suuren joukon ihmisiä, jolloin hieman tarkempien rajauksien tekeminen voi olla perusteltua. Tästä esimerkkinä toimii ”older people” (≥65)-termin tarkentaminen kahteen, ”youngest old” (60-69) ja ”oldest old” (≥80)-käsitteeseen [Heart and Kalderon, 2011]. Myös Godfrey ja Johnson [2009] jakavat omassa tutkimuksessaan aktiiviset vanhukset iän perusteella seuraavasti: ”younger old” (60-71) ja ”older old” (≥ 71). Kuten huomataan, näidenkään rajausten käyttö ei ole vakiintunutta, vaan ikähaarukat vaihtelevat tutkimuksesta toiseen.

Poliittiseen korrektiuteen pyrkiminen on viime vuosina lisääntynyt, ja se on ollut esillä myös pohdittaessa, millaisilla nimityksillä vanhuksiin tulisi erilaisissa tilanteissa viitata. Esimerkiksi vanhus-sanan käyttö on vähentynyt ja tilalle on tullut erilaisia enemmän ja vähemmän korrekteja kiertoilmauksia [Kaskiharju, 2004]. Analysoiduista artikkeleista tämä seikka näkyykin Salovaaran ja muiden [2010] tutkimuksessa, joka viittaa tutkittuun väestöryhmään vain iällä (55–65-vuotiaat). Henkilöitä ei siis pyritä kutsumaan millään nimityksellä ja näin ollen lukijalle ei myöskään pääse syntymään tutkimuksen kannalta vääränlaisia, ei-toivottuja, mielikuvia käytetystä käsitteestä ja tutkimuksen kohteina olevista ihmisistä. Onkin mielestäni mahdollista, että tulevaisuudessa tämänkaltainen viittaustekniikka tulee tutkimuksissa yleistymään.

4. Teknologia tutkimuksissa

Aktiivisten vanhusten suhtautumista teknologisiin laitteisiin ja tietoteknisiin sovelluksiin on vuosien varrella tutkittu monin erilaisin lähestymistavoin. Myös tutkimuksen aiheena oleva teknologia vaihtelee yksittäisestä teknisestä laitteesta laajempaan kokonaisuuteen, kuten laitteiden ja käyttöliittymien suunnitteluun. Nämä tutkituista artikkeleista löytyneet tulokset on koottu taulukkoon 3, jossa erilaiset teknologiat on lueteltu aakkosjärjestyksessä. Tämän lisäksi taulukkoon on koottu kolme artikkeleissa tasaisesti toistuvaa teemaa: yksinäisyys, motivaatio ja suunnittelu, sekä niiden esiintyminen kussakin julkaisussa (taulukko 3).

4.1 Millä tavoin ja mitä teknologiaa on tutkittu?

Käytettyjen tutkimusmenetelmien laajuus näkyy tämän katsauksen analyysiin päätyneissä julkaisuissa. Mukana on empiirisiä tutkimuksia, jossa tietyn laitteen tai sovelluksen käyttöä on havainnoitu pienessä ja rajatussa testihenkilöryhmässä. Esimerkkinä tästä toimii Fukudan [2011] neljä viikkoa kestänyt tutkimus, jossa seurattiin Nintendo-pelikoneen käytön vaikutusta kahdeksan vanhuksen elämään. Toisaalta analyysissa on myös laajempaan käyttäjäryhmään suuntautuvia kvantitatiivisia tutkimuksia, joiden tarkoituksena on saada kokonaisvaltaisempi kuva tutkitusta aiheesta, kuten esimerkiksi Internetin käyttötavoista tietyssä maassa [Chong and Theng, 2004].

Paitsi että tutkimusten menetelmät eroavat toisistaan, myös tutkittavissa teknologioissa on julkaisuiden välillä huomattavia eroja. Noin puolessa artikkeleista tutkittavaa teknologiaa ei määritellä tarkasti, vaan asian käsittely jätetään yleisemmälle tasolle. Saatetaan puhua vain teknisistä laitteista niitä tarkemmin nimeämättä. Esimerkiksi Eisma ja muut [2004] käsittelevät yleisesti teknologiaan liittyviä tuotteita sekä niiden suunnittelua ja käytettävyyttä vanhusten näkökulmasta.

Niistä artikkeleista, joissa tutkimuksen kohde on tarkemmin määritelty, kaikki käsittelevät varsin uutta teknologiaa. Tässä yhteydessä uudella teknologialla tarkoitetaan suhteessa melko lyhyen aikaa markkinoilla olleita tietokoneita, internetiä, kännyköitä ja digitaalisia televisioita. Voitaisiin myös sanoa, että niin kutsuttu uusi teknologia on sellaista, joka ei ole ollut saatavilla vielä tutkittavien omassa nuoruudessa tai siinä vaiheessa, kun he ovat olleet osa työelämää. Tämä tutkimuksen kohdistuminen uuteen teknologiaan näkyi jo tietokantahakuja tehdessä, sillä hakusanoista huomattavasti pidempään markkinoilla

ollut televisio tuotti paljon vähemmän tuloksia kuin esimerkiksi tietokone (taulukko 1).

Tutkimuksista noin puolet käsittelee selkeästi tietokoneita, internetin käyttöä ja ihmisen ja tietokoneen välistä vuorovaikutusta. Tämän lisäksi osa julkaisuista (noin 10%) puhuu teknisistä laitteista ja niiden suunnittelusta yleisemmällä tasolla. Lopuissa julkaisuissa aiheena ovat muun muassa kommunikointiin tarkoitetut laitteet (kuten kännykät), digitaaliset televisiot ja erilaiset pelisovellukset (taulukko 3). Kuten aiemmin jo mainittiin, tarkkojen rajavetojen tekeminen julkaisuiden välille on kuitenkin tässä suhteessa vaikeaa, sillä esimerkiksi teknisten laitteiden tutkiminen ja niiden suunnittelu pitää sisällään niin tietokoneet, televisiot kuin kännykätkin.

4. 2 Artikkeleiden tematiikka

Vaikka tutkitut artikkelit poikkeavatkin toisistaan, monissa niistä tulee esiin samoja teemoja aktiivisista vanhuksista ja heille suuntautuvan tekniikan suunnittelusta. Näitä toistuvia teemoja ovat aktiivisten vanhusten kokema yksinäisyys tai pelko syrjäytymisestä, vanhuksen oma motivaatio uuden oppimisessa sekä heille tarkoitettujen laitteiden suunnittelu ja siinä huomioon otettavat seikat.

Yhteiskunnan teknologisoituminen on ollut nopeaa ja tästä syystä on ollut tärkeää tarkastella, kuinka ihmiset ovat pysyneet kehityksessä mukana. Kehityksen ulkopuolelle jäämisen suurimpia ongelmia saattavat jossain tapauksessa olla yksinäisyyden kokeminen ja yhteiskunnasta syrjäytyminen. Suurimmassa syrjäytymisvaarassa ovat usein erityisryhmiin kuuluvat ihmiset, joilla on erilaisia, muusta kansasta poikkeavia tarpeita. Vanhusten kohdalla nämä tarpeet liittyvät LeRougen ja muiden [2011] mukaan ikääntymisen mukanaan tuomiin psyykkisiin ja fyysisiin muutoksiin. Ikääntymisen myötä ihminen myös todennäköisemmin kokee yksinäisyyttä [Toepoel, 2012]. Erilaisten teknologisten sovellusten ja laitteiden käyttäminen usein vähentääkin tätä yksinäisyyden ja eristäytyneisyyden tunnetta [Sayago and Blat, 2010].

Toinen julkaisuissa toistuvista teemoista on vanhusten oma aktiivisuus ja motivaatio oppimisen kannalta. Motivaation merkitys on suuri, jos halutaan, että uuden teknologian käyttöönotto on onnistunutta [Nora *et al.*, 2011]. Salovaara ja muut [2012] ovat puolestaan havainneet, että ihmiset ovat usein halukkaampia oppimaan uusista teknologioista, jos he kokevat hyötyvänsä siitä omassa arkipäiväisessä elämässään. Usein esimerkiksi teknologian avulla ta-

pahtuva yhteydenpito läheisiin ihmisiin koetaan motivoivaksi tekijäksi [Sayago *et al.*, 2011]. Useat tutkimukset sivuavatkin sekä yksinäisyyttä että motivaatiota, sillä yksinäisyyden kokeminen toimii usein rohkaisevana ja motivoivana tekijänä uuden teknologian käyttöä opeteltaessa.

Vaikka aktiivisilla vanhuksilla usein olisikin motivaatiota opetella tietotekniikan käyttöä, heidän sopeutumisensa uuden teknologian käyttöönottoon on edelleen rajoittunutta [Heart and Kalderon, 2011]. Eräs suurimmista syistä tähän voi olla se, ettei vanhuksia ja heidän tarpeitaan oteta nykyisin riittävästi mukaan laitteiden varhaisessa suunnitteluvaiheessa [LeRouge *et al.*, 2011]. Le Rougen ja muiden [2011] mukaan suunnittelun tekevät yleensä nuoremmat ja osittain hyväkuntoisemmat ihmiset, jotka eivät kärsi esimerkiksi ikänäöstä tai heikentyneestä kuulosta. Tällöin lopulliset tuotteet eivät välttämättä sovellukaan niille, joille se on tarkoitettu. Song ja Lee [2008] korostavatkin sitä, että kaikkein tärkein vaihe tekniikan suunnittelussa olisi todellisten käyttäjätarpeiden huomioon ottaminen.

Monet tutkimukset tuovat esiin ajatuksen siitä, että samat tuotteet pitäisi pyrkiä suuntaamaan yhä useammalle eri käyttäjäryhmälle. Kuten Emiliani [2002] tuo esiin, nykyisin teknologinen ympäristö on hyvin monimuotoinen ja kirjava. Tämän vuoksi niin kutsuttu idea ”keskivertokäyttäjästä” ei enää suunnittelussa toimi, vaan kaikkien potentiaalisten käyttäjien vaatimukset tulisi ottaa huomioon. Hieman samaa ajatusta kannattavat myös Goodman-Deane ja muut [2009], sillä heidän mukaansa tutkimusta vanhempien ihmisten tarpeista tulisi tehdä, jotta valtavirralle suunnatut tuotteet voitaisiin muokata vastaamaan myös heidän kysyntäänsä. Jos tuotteen suunnittelussa on otettu huomioon erityisryhmien tarpeet, se soveltuu yleensä silloin hyvin myös muille, tavallisille käyttäjäryhmille [Zajicek, 2006].

Authors	Year	Technology	Loneliness, Isolation	Motivation	Design
Dickinson, Gregor	2006	Computers	X	-	-
Esteller-Curto, Escuder-Mollon	2012	Computers	-	X	-
Nora, Razaka, Abdullaha *et al.*	2011	Computers	-	X	-
Slegers, van Boxtel, Jolles	2012	Computers	X	X	-
Sayago, Blat	2010	Computers, e-mail	X	X	X
Zajicek	2006	Computers, HCI	-	-	X
Chong, Theng	2004	Computers, internet	X	X	X
Niehaves, Plattfault	2010	Computers, internet	-	X	-
Ushida, Hata, Matsuura *et al.*	2001	Computers, mobile phones	-	-	-
Sayago, Sloan, Blat	2011	Computer mediated communication (CMC), ICT	X	X	X
Eisma, Dickinson, Goodman	2004	Design and usability of technology related products.	X	X	X
Rice, Alm	2007	Digital television (DTV)	-	-	X
Rice, Carmichael	2011	Digital television (DTV)	X	X	X
Escuder-Mollon	2012	Education (also ITC)	X	X	-
Kang, Kim, Kim	2009	Gate–ball game	-	X	X
Goodman-Deane, Keith, Whitney	2009	HCI, ICT	-	-	X
Goodman, Lundell	2005	HCI, user interface	X	X	X
Heart, Kalderon	2011	ICT	-	-	-
Salovaara *et al.*	2010	ICT (computers, mobile phones)	X	X	X
Eronen	2006	Interactive television (iTV)	X	X	X
Kurniawan, King, Evans *et al.*	2006	Internet	-	-	X
Peacock, Künemund	2007	Internet	-	-	X
Dickinson, Newell, Smith *et al.*	2005	Internet, e-mail	-	-	X
Godfrey, Johnson	2009	Information access	-	-	X
Emiliani	2002	Information technology	-	-	X
Toepoel	2012	Leisure activities (refers to television, computer)	X	-	-
Kurniawan	2008	Mobile phones	-	-	X
Von Bruhn Hinné, Keates	2011	Motion sensitive remote control devices	-	-	X
Tielen	1998	New information technology	X	X	X
Fukuda	2011	Nintendo	-	X	-
Blythe, Monk, Doughty	2005	Technology that can support independent living	-	-	X
Pieper, Hermsdorf	1997	Telecommunication technologies	-	-	X
Song, Lee	2008	Universal design	-	-	X
LeRouge, Ma, Sneha *et al.*	2011	User-Centered Design	-	-	X

Taulukko 3: Teknologia tutkimuksissa

5. Pohdintaa

Tutkimuksesta nousee esiin muutama kysymys, joita seuraavassa tarkastelen. Esittelen näiden pohjalta myös muutaman ehdotuksen jatkotutkimuksen aiheeksi.

Kuten aiemmin teknologiaosaa käsiteltäessä toin esiin, työhön mukaan päätyneet artikkelit käsittelivät yksinomaan varsin uutta teknologiaa. Syitä tämän analysointiin päätyneen tekniikan yksipuolisuudelle voitaneen lähteä etsimään niin hakusanoista kuin analysoitujen tutkimusten tavoitteistakin.

Tekniikkaa koskevista hakusanoista technology, computer, ICT ja HCI tuottivat selkeästi eniten tuloksia (taulukko 1). Tulosten runsaus johti myös siihen, että samoja hakusanoja oli helppo käyttää uudestaan yhdistäen niitä aktiivisiin vanhuksiin viittaaviin eri nimityksiin. Samaa ei tullut tehtyä muutaman muun käsitteen, kuten televisio ja radio, kohdalla, sillä ne eivät ensimmäisissäkään hauissa juuri tuottaneet tuloksia. Tästä herää kysymys, olisiko oikeanlaiset hakusanat löytämällä ollut mahdollista tavoittaa tutkimuksia myös vanhempaan teknologiaan liittyen.

Epäilen kuitenkin, ettei tulosten painottuminen johtunut yksinomaan käytetyistä käsitteistä tai niiden yksipuolisuudesta. Koska uusien laitteiden ja sovellusten käyttö ei ole vielä täysin vakiintunutta, niihin liittyy osittain tuntemattomia ilmiöitä ja selvitystä vaativia seikkoja. Näitä tutkijoiden on mielekästä pyrkiä saamaan selville, sillä tällöin pystytään ottamaan tekniikan nykyiset puutteet huomioon uutta suunnittelua ja kehitystä tehdessä. Julkaisuiden painottuminen uusien asioiden tutkimiseen on siis vain luonnollista.

Teknologian tutkimisen painottuminen uusiin laitteisiin näkyy myös seuraavasti. Vaikka ikääntynyt ei käyttäisi arkielämässään tietokonetta tai kännykkää, saattaa hän silti aktiivisesti käyttää itselleen tutuksi tulleita laitteita ja koneita. Tällaisia kokonaisvaltaisia tutkimuksia aktiivisten vanhusten käyttämistä teknisistä laitteista ja sovelluksista ei juuri ole saatavilla.

Tämä työ pohjautui englanninkieliseen aineistoon ja näin ollen myös tutkittavat käsitteet ovat englanniksi. Harkitsin myös perehtyväni siihen, minkälaisia nimityksiä aktiivisista vanhuksista suomenkielisissä tutkimuksissa käytetään. Luovuin tästä ajatuksesta kuitenkin lähinnä sen vuoksi, ettei aihepiiristä löytynyt kovinkaan paljon tieteellisiä artikkeleita. Tämän tutkimuksen konteksti, eli tekniikka ja teknologia, varmasti osittain aiheutti tämän materiaalin vähäisen määrän. Jatkotutkimusta käsitteiden käytöstä olisikin mahdollista tehdä

laajentamalla tutkimus koskemaan kaikenlaista aktiivisista vanhuksista tehtyä tieteellistä tutkimusta.

6. Yhteenveto

Tässä tutkielmassa haluttiin selvittää, minkälaisia käsitteitä aktiivisista vanhuksista tutkimuksissa käytetään ja onko näiden eri nimityksien välille mahdollista löytää jotain, joka erottaa ne toisistaan. Työn toinen tarkoitus oli perehtyä siihen, millaista teknologiaa aktiiviset vanhukset tutkimusten mukaan käyttävät. Tutkimuksen metodina toimii kirjallisuuskatsaus ja lopullinen analyysi koostuu 35 artikkelista.

Aktiivista vanhuutta koskevia käsitteitä tarkastellessa havaittiin, että käytetyt termit ja niiden määritykset eivät ole keskenään täysin vertailtavissa, vaan ne vaihtelevat tutkimuksesta toiseen. Näin ollen samalla käsitteellä voidaan eri julkaisuissa viitata ihmisiin, joiden iät ja toimintakyvyt poikkeavat toisistaan. Yleisimmiksi käsitteiksi paljastuivat vanhus, vanhemmat ihmiset ja seniorit. Näillä kaikilla viitataan aktiivisiin vanhuksiin, mutta niitä määriteltäessä alimmat ikärajat vaihtelevat 50-vuotiaista 70-vuotiaisiin. Ylärajaa ei yhtä usein mainita, mutta määritellyissä se vaihtelee 64-vuotiaista 89-vuotiaisiin.

Teknologiaa tutkittaessa huomio kiinnittyi siihen, kuinka sen tutkiminen painottuu uuteen tekniikkaan. Erityisen tutkittuja teknologioita tuntuvat olevan erilaiset kommunikointiin soveltuvat teknologiat, kuten tietokoneet ja kännykät. Keskeisiä artikkeleista esille nousseita teemoja ovat aktiivisten vanhusten kokema yksinäisyyden tunne ja mahdollinen syrjäytyminen yhteiskunnasta, motivaation merkitys teknologian käyttöä opeteltaessa sekä teknologian suunnittelussa huomioon otettavat seikat.

Viiteluettelo

[Blythe *et al.*, 2005] Mark A. Blythe, Andrew F. Monk and Kevin Doughty, Socially dependable design: the challenge of ageing populations for HCI. *Interacting with Computers* **17**, 6 (Dec. 2005), 672-689.

[Chong and Theng, 2004] Seck-Pin Chong and Yin-Leng Theng, A study of webbased information needs of senior citizens in Singapore. *Lecture Notes in Computer Science* **3196** (2004), 16-33.

[Dickinson and Gregor, 2006] Anna Dickinson and Peter Gregor, Computer use has no demonstrated impact on the well-being of older adults. *International Journal of Human-Computer Studies* **64,** 8 (Aug. 2006), 744-753.

[Dickinson *et al.*, 2005] Anna Dickinson, Alan F. Newell, Michael J. Smith and Robin L. Hill, Introducing the Internet to the over-60s: developing an email system for older novice computer users. *Interacting with Computers* **17**, 6 (Dec. 2005), 621-642.

[Eisma *et al.*, 2004] R. Eisma, A. Dickinson, J. Goodman, A. Syme, L. Tiwari and A. F. Newell, Early user involvement in the development of information technology related products for older people. *Universal Access in the Information Society* **3**, 2 (2004), 131-140.

[Emiliani, 2002] Pier Luigi Emiliani, New technologies and services for disabled and elderly people in the emerging information society. *Lecture Notes in Computer Science* **2398** (2002), 119-152.

[Eronen, 2006] Leena Eronen, Five qualitative research methods to make iTV applications universally accessible. *Universal Access in the Information Society* **5**, 2 (Aug. 2006), 219-238.

[Escuder-Mollon, 2012] Pilar Escuder-Mollon, Modelling the impact of lifelong learning on senior citizens' quality of life. *Procedia – Social and Behavioral Sciences* **46** (2012), 2339-2346.

[Esteller-Curto and Escuder-Mollon, 2012] Roger Esteller-Curto and Pilar Escuder-Mollon, Non-practical ICT courses for seniors for a comprehensive involvement to provide a global understanding of the knowledge society. *Procedia – Social and Behavioral Sciences* **46** (2012), 2356-2361.

[Fukuda, 2011] Ryoko Fukuda, Affective technology for older adults: does fun technology affect older adults and change their lives? *Lecture Notes in Computer Science* **6766** (2011), 140-148.

[Godfrey and Johnson, 2009] Mary Godfrey and Owen Johnson, Digital circles of support: meeting the information needs of older people. *Computers in Human Behavior* **25**, 3 (May, 2009), 633-642.

[Goodman and Lundell, 2005] Joy Goodman and Jay Lundell, HCI and the older population. *Interacting with Computers* **17**, 6 (Dec. 2005), 613-620.

[Goodman-Deane *et al.*, 2012] Joy Goodman-Deane, Suzette Keith and Gill Whitney, HCI and the older population. *Universal Access in the Information Society* **8**, 1 (Apr. 2009), 1-3.

[Heart and Kalderon, 2011] Tsipi Heart and Efrat Kalderon, Older adults: Are they ready to adopt health-related ICT?, *International Journal of Medical Informatics* (Apr. 2011).

[Kang *et al.*, 2009] Kyung-Kyu Kang, Jung-A Kim and Dongho Kim, Development of a sensory gate–ball game system for the aged people. *The Visual Computer* **25**, 12 (Dec. 2009), 1073-1083.

[Kaskiharju, 2004] Eija Kaskiharju, Vanhus, ikäihminen vai seniorikansalainen? *Gerontologia* **4** (2004), 277-281.

[Kurniawan, 2008] Sri Kurniawan, Older people and mobile phones: a multi-method investigation. *International Journal of Human-Computer Studies* **66**, 12 (Dec. 2008), 889-901.

[Kurniawan *et al.*, 2006] S. H. Kurniawan, A. King, D. G. Evans and P. L. Blenkhorn, Personalising web page presentation for older people. *Interacting with Computers* **18**, 3 (May. 2006), 457-477.

[Lee and King, 2003] Rebecca F. Lee and Abby C. King, Discretionary time among older adults: how do physical activity promotion interventions affect sedentary and active behaviors? *Annals of Behavioral Medicine* **25**, 2 (Apr. 2003), 112-119.

[LeRouge *et al.*, 2011] Cynthia LeRouge, Jiao Ma, Sweta Sneha and Kristin Tolle, User profiles and personas in the design and development of consumer health technologies. *International Journal of Medical Informatics*, April (2011), 1-18.

[Niehaves and Plattfault, 2010] Bjoern Niehaves and Ralf Plattfaut, What is the issue with internet acceptance among elderly citizens? Theory development and policy recommendations for inclusive e-government. *Lecture Notes in Computer Science* **6228** (2010), 275-288.

[Nora *el al.*, 2011] Nor Fariza Mohd. Nora, Norizan Adbul Razaka, Mohd Yusof Abdullaha, Jalaluddin Abdul Maleka and Ali Salmana, Empowering marginalized community with an innovative technology. *Procedia - Social and Behavioral Sciences* **15** (2011), 3374-3378.

[Peacock and Künemund, 2007] Sylvia E. Peacock and Harald Künemund, Senior citizens and internet technology. *European Journal of Ageing* **4**, 4 (Dec. 2007), 191-200.

[Pieper and Hermsdorf, 1997] Michael Pieper and Dirk Hermsdorf, BSCW for disabled teleworkers: usability evaluation and interface adaptation of an internet-based cooperation environment. *Computer Networks and ISDN Systems* **29** (1997), 1479-1487.

[Rice and Alm, 2007] Mark Rice and Norman Alm, Sociable TV: exploring user-led interaction design for older adults. *Lecture Notes in Computer Science* **4471** (2007), 126-135.

[Rice and Carmichael, 2011] Mark Rice and Alex Carmichael, Factors facilitating or impeding older adults' creative contributions in the collaborative design of a novel DTV-based application. *Universal Access in the Information Society*, 24 November 2011.

[Salovaara *et al.*, 2010] Antti Salovaara, Asko Lehmuskallio, Leif Hedman, Paula Valkonen and Jaana Näsänen, Information technologies and transitions in the lives of 55–65-year-olds: The case of colliding life interests. *International Journal of Human-Computer Studies* **68**, 11 (Nov., 2010), 803-821.

[Sayago and Blat, 2010] Sergio Sayago and Joseph Blat, Telling the story of older people e-mailing: an ethnographical study. *International Journal of Human-Computer Studies* **68**, 1-2 (Jan.-Feb. 2010), 105-120.

[Sayago *et al.*, 2011] Sergio Sayago, David Sloan and Joseph Blat, Everyday use of computer-mediated communication tools and its evolution over time: an ethnographical study with older people. *Interacting with Computers* **23**, 5 (Sep. 2011), 543-554.

[Slegers *et al.*, 2012] Karin Slegers, Martin P.J. van Boxtel and Jelle Jolles, Computer use in older adults: Determinants and the relationship with cognitive change over a 6 year episode. *Computers in Human Behavior* **28**, 1 (Jan., 2012), 1-10

[Song and Lee, 2008] Kyohyun Song and Seongil Lee, Mapping user accessibility needs systematically to universal design principles. *Lecture Notes in Computer Science* **5068** (2008), 446-456.

[STM, 1999] Sosiaali- ja terveysministeriö STM, Vanhusbarometri, 1999. http://pre20031103.stm.fi/suomi/pao/julkaisut/vbaro/vbyhtve.htm. Viitattu 24.10.2012.

[SVT, 2012] Suomen virallinen tilasto SVT, Väestöennuste, 2012. Helsinki: Tilastokeskus. http://www.stat.fi/til/vaenn/2012/vaenn_2012_2012-09-28_tie_001_fi.html. Viitattu 12.10.2012.

[SVT, 2011] Suomen virallinen tilasto SVT, Tieto- ja viestintätekniikan käyttö, 2011. Helsinki: Tilastokeskus. http://www.stat.fi/til/sutivi/2011/sutivi_2011_2011-11-02_tie_001_fi.html. Viitattu 25.10.2012.

[Tielen, 1997] Ger Tielen, Integrating senior citizens into the information society. *Ageing International* **24**, 2-3 (1997-1998), 143-153.

[Toepoel, 2012] Vera Toepoel, Ageing, leisure, and social connectedness: how could leisure help reduce social isolation of older people? *Social Indicators Research* (June 2012).

[Ushida *et al.,* 2001] Hayato Uchida, Yutaka Hata, Shinro Matsuura, Tadahiro Tsuchikawa, Yoshio Morotomi and Hideyasu Aoyam, Rough set based knowledge discovery of interface for the internet usage among Japanese elderly women. *Lecture Notes in Computer Science* **2206** (2001), 749-754.

[von Bruhn Hinné and Keates, 2011] Thomas von Bruhn Hinné and Simeon Keates, Using motion-sensing remote controls with older adults. *Lecture Notes in Computer Science* **6766** (2011), 166-175.

[Zajicek, 2006] Mary Zajicek, Aspects of HCI research for older people. *Universal Access in the Information Society* **5**, 3 (Nov. 2006), 279-286.

Liikeparallaksin toteutus ja erot stereoskopiaan 3D-sovelluksissa

Tomi Fagerlund

Tiivistelmä.
Tämä tutkielma esittelee tekniikan, joka käyttää liikeparallaksia tuottamaan kolmiulotteisen syvyyshavainnon tavalliselle tietokoneen näytölle tai muulle ruudulle, kuten videoprojektorilla heijastetulle kankaalle. Liikeparallaksin toteuttamiseen liittyviä seikkoja kuvataan ja sille esitellään sopivia sovellusalueita sekä vertaillaan vielä nykyisin käytetympään tapaan 3D-näkymien tuottamiseen, joka perustuu stereoskopiaan. Molempien menetelmien teknologioista annetaan myös esimerkkejä.

Avainsanat ja -sanonnat: 3D-näyttö, liikeparallaksi, stereoskopia
CR-luokat: I.3.3., I.3.7., I.3.8

1. Johdanto

Havaitsemme meitä ympäröivän maailman jatkuvasti kolmiulotteisena, minkä vuoksi kolmiulotteisesta havainnosta on myös hyötyä tietokoneympäristöissä ja käyttöliittymissä. Kolmiulotteisia pelejä ja muita sovelluksia on kehitetty jo usean vuosikymmenen ajan, mutta kaksiulotteisella näytöllä 3D-näkymät näyttävät litteiltä ilman syvyysvaikutelmaa, vaikka näkymä itsessään sisältäisi informaation kolmannesta ulottuvuudessa ja sen renderöimiseen käytettäisiin tehokkasta näytönohjainta ja uusinta teknologiaa. Aivomme rakentavat kolmiulotteisen mallin ympäristöstä useiden aistihavaintojen perusteella. Tärkein aisti on tietenkin näköaisti, vaikka myös kuulo- ja tuntoaistilla on merkitystä. Näköaistin kautta saamme useita eri vihjeitä ympäristössä olevien esineiden etäisyyksistä. Vihjeistä tärkeimmät ovat stereoskopia ja liikeparallaksi.

3D-TV:t ja -elokuvat ovat jälleen alkaneet yleistyä muutaman viime vuoden aikana. Kehittynyt teknologia tarjoavaa uusia ja aikaisempaa parempia mahdollisuuksia 3D-sovellusten kehittämiseen. Myös lisätty- ja virtuaalitodellisuusympäristöt ovat tulossa kotikäyttäjille. Valtaosa nykyisistä 3D-TV:istä ja -elokuvista käyttävät stereoskopiaa kolmiulotteisen havainnon tuottamiseen. Liikeparallaksin hyödyntäminen on toistaiseksi jäänyt vähemmälle, mutta viime vuosina kuluttajamarkkinoille tulleet uudet laitteistot voivat mahdollistaa liikeparallaksin käytön laajemmin 3D-sovelluksissa.

Liikeparallaksi pelkästään ei täysin sovellu kaikkiin käyttötapauksiin, mutta siihen perustuen voidaan kehittää uusia sovelluskohteita. Ilmiselvä parallaksin rajoite on, että yhtä katselijaa kohti tarvitaan yksi näyttö. Esimerkiksi suurella yleisöllä katsottavat elokuvat eivät voi hyötyä liikeparallaksista. Liikeparallaksilla saadaan kuitenkin monissa tilanteissa etua stereoskopiaan verrattuna.

Tässä tutkielmassa tarkastellaan liikeparallaksin hyödyntämistä tietokonesovelluksissa ja vertaillaan sitä toistaiseksi käytetympään stereoskopiaan. Aluksi esitellään molempia tärkeimmistä kolmiulotteisen havainnon tuottavista näkövihjeistä ja niiden käyttämistä ja tutkimuksia tietokonesovelluksissa sekä paneudutaan esittelemään liikeparallaksin toteuttamista sovelluskehittäjän näkökulmasta. Lopuksi vertaillaan, minkälaisiin sovellusalueisiin kaksi esiteltyä tekniikkaa soveltuvat parhaiten.

2. 3D-näyttöjen teknologiat

3D-näytöt mielletään usein passiiviseen katseluun tarkoitetuiksi laitteiksi. Nykyisin lähes kaikki 3D-näytöt, kuten 3D-elokuvat ja -TV:t perustuvat binokulaariseen eroavuuteen eli stereoskopiaan. Stereoskopiaan käytetyistä teknologioista johtuen katselukulma, jossa 3D-efekti on mahdollista nähdä, on usein kapea, eikä näkymän perspektiivi muutu, vaikka näkymää katsottaisiin eri kohdista [Urey et al., 2011]. Katselijalle ei jää tilaa liikkua, eikä liikkumisella saavuteta mitään lisäarvoa näkymän katselemisessa. Uusilla teknologioilla on kuitenkin mahdollista tuottaa 3D-efekti, joka mahdollistaa myös perspektiivin vaihtumisen, ja siten uudet teknologiat avaavat uusia mahdollisuuksia sovelluskehittäjille.

Stereoskooppinen havainto syntyy, kun ympäristöä havainnoidaan kahdesta erillään olevasta pisteestä. Molempien havaintopisteiden saamat kuvat yhdistetään yhdeksi kuvaksi, johon saadaan informaatio objektien etäisyyksistä alkuperäisten kuvien eroavuuksien perusteella. Liikeparallaksi-ilmiö taas syntyy, kun havaintopiste muuttuu eli katselija liikkuu samalla kun havaitsee ympäristöä. Eri kohdista saatujen kuvien eroavuuksien perusteella myös tässä tapauksessa saadaan informaatio etäisyyksistä. Pohjimmiltaan liikeparallaksi ja stereoskopia perustuvat samaan ilmiöön, mutta käytännössä ne toimivat hyvin eri tavalla.

Molempia näistä kahdesta ilmiöstä käytetään 3D-näyttöjen teknologioiden perustana. Stereoskoppinen menetelmä vaatii lähes aina erityisen näytön, kuten uudet 3D-televisiot, ja usein myös puettavat silmälasit, mutta liikeparallaksilla syvyysvaikutelma on mahdollista luoda myös tavalliselle tietokoneen näytölle tai muulla näyttölaitteella ilman puettavia lisälaitteita. Sovelluskehittäjän näkö-

kulmasta liikeparallaksi on monessa tapauksessa helpompi ja halvempi toteuttaa. Kuluttajamarkkinoilla on tarjolla valmiita paketteja stereoskoopiselle 3D-efektille, kuten *NVidia*n *3D Vision Kit*, mutta nämä yhdessä vaadittavan näytön kanssa maksavat huomattavasti enemmän kuin liikeparallaksiin tarvittavat laitteistot [Li et al., 2012].

Syvyyden havaitsemista tietokoneympäristöissä on tutkittu kattavasti etenkin virtuaalistodellisuussovellusten yhteydessä [Jones et al., 2008; Naepflin and Menozzi, 2001]. Tutkimuksissa on havaittu, että virtuaalitodellisuussovellusten käyttäjät usein aliarvioivat etäisyyksiä. Jones ja muut [2008] tutkivat myös liikeparallaksin vaikutusta etäisyyksien arviointiin. Vastoin odotuksia heidän tulosten mukaan liikeparallaksi lisäsi aliarviointia. Jonesin ja muiden [2008] kokeessa tosin käytettiin päähän puettavaa näyttölaitetta, jonka massa vaikutti katselijan havainnointikykyyn ja mahdollisesti vaikutti etäisyyksien hahmottamiseen.

Tästä huolimatta muissa tutkimuksissa on havaittu, että liikeparallaksiefekti antaa etenkin kolmiulotteisen näkymän hahmottamisessa todentuntuisemman ja tarkemman syvyysvaikutelman [Naepflin and Menozzi, 2001]. Liikeparallaksin avulla on myös mahdollista 3D-näkymän todellinen havaitseminen useasta kulmasta, eli katsoja voi nähdä, miltä esineet näyttävät toisesta kulmasta ja esineiden taakse piilotettuja esineitä. Yhteen näkymään saadaan siis enemmän informaatiota ja tämä informaatio voidaan järjestää tasoihin, jotka ovat käyttäjälle intuitiivisia. Lisäksi liikeparallaksi on todettu miellyttävämmäksi [Li et al., 2012]. Tutkimustulokset liikeparallaksin ja stereskopian eroista syvyyshavainnoinnin kannalta eivät siis ole yksimielisiä. Liikeparallaksin toteutusta ja hyötyjä käsitellään laajemmin myöhemmin.

Seuraavaksi tarkastellaan stereoskopian ja liikeparallaksin toteuttamisen menetelmien pääpiirteisiä eroja.

2.1. Stereoskopia

Käytetympi ja tunnetumpi menetelmä syvyyshavaintojen tuottamiseen on stereoskooppinen efekti. Sovelluksissa tämä menetelmä perustuu kahden samasta 3D-näkymästä hieman eri kulmista otetun kuvan suodattaminen katselijan eri silmille.

Stereoskopian käyttö vaatii lähes aina jonkinlaisten lasien käyttöä. Erityisillä näyttölaitteilla on tosin mahdollista luoda stereoskooppinen vaikutelma ilman laseja. Näitä autostereoskooppisia menetelmiä esitellään myöhemmin. Seuraavaksi tarkastellaan erilaisia stereoskooppiefektin toteutustekniikoita.

Stereoskooppiseen menetelmään perustuvat sovellukset voidaan toteuttaa karkeasti määriteltynä kolmella eri tekniikalla [Urey et al., 2011].

Värikanavoitu lähestymistapa on tekniikan vanhin. Siinä käytetään silmälaseja erottelemaan kuvat vastavärikanavoinnilla. Yleisimmin käytetty kanavointi on punainen vasemmalle silmälle ja sinivihreä oikealle. Tämä metodi ymmärrettiin jo 1800-luvulla [Urey et al., 2011]. Haittapuolena on värien menettäminen näkymästä ja epätäydellinen kuvien erottelu oikealle ja vasemmalle silmälle [Urey et al., 2011]. Menetelmä on kuitenkin hyvin halpa ja sen voi toteuttaa millä tahansa värien toistamiseen pystyvällä näytöllä. Myös 3D-rajapinnoissa on valmiina toiminnallisuus värikanavoinnin toteuttamiseen valmiiseen 3D-sovellukseen [Li et al., 2012]. Sovelluskehittäjien siis ei tarvitse tehdä suuria muutoksia ohjelman lähdekoodiin tämän tekniikan toteuttamiseksi.

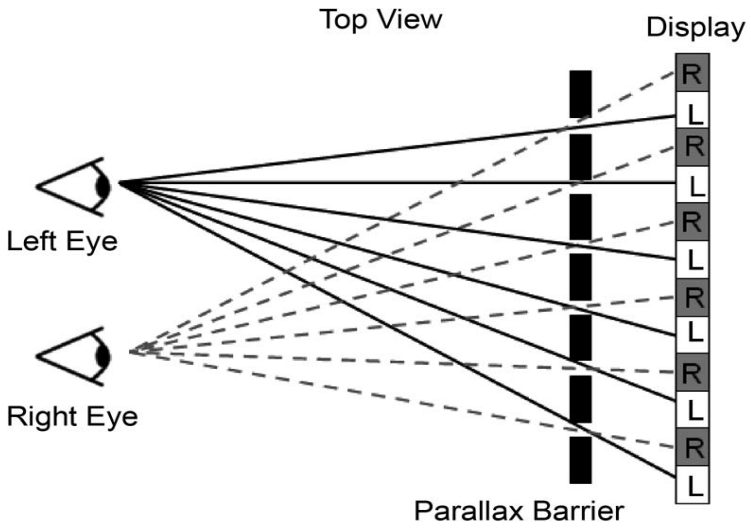
Polarisaatiokanavoidun tekniikan perustana on valon polarisaation hyödyntäminen. Molempien stereokuvien polarisaatio käännetään kohtisuoraan toiseen nähden. Ympyräpolarisaatio mahdollistaa lineaarista laajemman katselukulman ilman kuvien vuotamista toiselle silmälle [Urey et al., 2011]. Katselijan tarvitsee käyttää laseja, jotka erottelevat polarisoidun valon ja päästävät vain oikean kuvan läpi. Katsomiseen tarvitaan myös kaksi projektoria, jotka voivat lähettää kohtisuoraan polarisoitunutta valoa toisiinsa nähden. Lisäksi heijastuspinnan tulee pystyä säilyttämään polarisaation tila [Urey et al., 2011]. Tämä tekniikka siis vaatii kalliiden laitteiden käyttöä.

Aikakanavoidussa menetelmässä oikea ja vasen kuva näytetään vuorotellen yhdellä näytöllä nopealla kuvanopeudella. Usein nopeutena käytetään 120 Hz [Urey et al, 2011]. Katsojan tarvitsee käyttää aktiivilaseja, jotka synkronisesti näytön kanssa sulkevat oikean puolen, kun ruudulla näkyy vasen kuva, ja päinvastoin. Tämän menetelmän haittapuolina on kalliiden ja mahdollisesti epämukavien paristokäyttöisten aktiivilasien käyttäminen ja tarve ylimääräiselle datakaistalle johtuen nopeasta kuvanopeudesta [Urey et al., 2011]. Lisäksi tavalliset näytöt eivät pysty vaadittuun kuvanopeuteen.

Stereoskooppisen efektin voi tuottaa myös ilman laseja. Tällöin siitä käytetään nimitystä autostereoskopia. Autostereoskooppiset näytöt itsessään perustuvat jonkinlaiseen tekniikkaan, jolla eri kuvat lähetetään eri silmille. Usein näissä laitteissa katselukulma jää kapeaksi [Urey et al., 2011]. Autostereoskooppiset näytöt jaetaan kaksi- ja moninäkymäisiksi. Kaksinäkymillä tuotetaan vain yksi stereokuva yhdelle katsojalle, moninäkymillä tuotetaan useita stereokuvia useille katselijoille. Seuraavaksi tarkastellaan muutamia autostereoskooppisia tekniikoita.

Kaksinäkymänäytöillä tuotetaan kaksi kuvaa, jotka muodostavat yhden stereoparin yhdelle katsojalle joko yhteen tiettyyn pisteeseen ruudun edessä tai sama kuva voidaan toistaa useisiin kohtiin [Urey et al., 2011], mutta katsojan silmien täytyy olla oikeassa kohdassa ideaalilla etäisyydellä stereoskooppisen kuvan havaitsemiseksi. Kaksinäkymään voidaan myös lisätä päänseuranta, jolloin katselupiste ei ole niin tarkasti rajattu.

Parallaksisulkunäyttö on eräs kaksinäkymänäytön toteutustekniikka. Näytön LCD-elementtien eteen kiinnitetään pystysuorista esteistä koostuva sulkupaneeli, joka erottelee alla olevat kuvapisteet oikealle ja vasemmalle silmälle. Kuvassa 1 on kaavio näytön toiminnasta.



Kuva 1. Parallaksisulkunäytön toiminnan periaate [Urey et al. 2011]

Kaksinäkymänäyttö voidaan myös toteuttaa linsseillä. Linssijärjestelmään perustuva kaksinäkymänäyttö käyttää kaareutuvia linssejä taittamaan projektoreista lähetettyä valoa parallaksisulkujen avulla katselijan silmille [Urey et al., 2011].

Moninäkymäiset näytöt projisoivat useita stereokuvapareja moneen kohtaan näytön edessä. Näkymää voi siis katsoa useasta kulmasta, mutta näissä järjestelmissä kuvaparien määrä ei ole riittävä jatkuvan parallaksiefektin saavuttamiseksi [Urey et al., 2011]. Useampi katselija voi kuitenkin katsoa samaa 3D-näkymää samanaikaisesti.

Moninäkymänäyttöihin voidaan myös lisätä käyttäjien pään seuranta, jolloin saavutetaan laajempi katselukulma ja katselupisteet eivät enää ole kiinnitetty tiettyyn kohtaan näytön edessä. Urey ja muut [2011] kuvaavat neljä tällaista perustekniikkaa: Fresnel-linssi, linssimäinen, projektio ja parallaksisulku. Nämä toimivat esimerkiksi erilaisten linssien ja parallaksisulkujen avulla,

mutta pääasiallinen ero liikeparallaksimenetelmään on, että näissä lähetetään yksi stereokuvapari katselijoiden silmille, eivätkä näytöt kykene tuottamaan sulavaa parallaksiefektiä.

Valokenttänäytössä jokainen kuvapiste voi lähettää useita valonsäteitä eri kulmissa. Periaatteessa tällä tekniikalla yhdessä käyttäjän seuraamisen kanssa pystytään tuottamaan täydellinen 3D-efekti usealle käyttäjälle yhtä aikaa. Urey ja muut [2011] esittelevät muutamia kokeellisia esimerkkejä tällaisista valokenttänäytöistä, mutta tämä tekniikka vaatii vielä paljon kehittämistä ennen kuin sillä voidaan kehittää sovelluksia kuluttajamarkkinoille.

Lisäksi yksi stereoskopiaan perustuva 3D-näyttölaite on näyttölasit. Lasit itsessään sisältävät kaksi erillistä näyttöä, jotka suoraan näyttävät eri kuvat oikealle ja vasemmalle silmälle. Laseilla voidaan tuottaa täysin immersoiva virtuaalitodellisuusympäristö, mutta toistaiseksi nämä laitteet eivät ole yleistyneet niiden epäkäytännöllisyyden ja korkean hinnan takia [Li et al., 2012]. Tätä tekniikkaa kuitenkin kehitetään jatkuvasti eteenpäin, ja uudet teknologiat, kuten *Google*n *Glass* ja *Oculus VR*:n kaltaiset projektit saattavat tulevaisuudessa tuoda tämän tekniikkan päivittäiseen käyttöön.

2.2. Liikeparallaksi

Liikeparallaksi-ilmiö syntyy, kun havaitsija tarkastelee eri etäisyyksillä olevia esineitä ja liikkuu samalla eli vaihtaa havaintopistettä. Havaitsija liikkuu suhteessa enemmän lähempänä oleviin objekteihin, jolloin lähempänä olevat objektit näyttävät liikkuvan kauempana olevien edessä. Ilmiön voi helposti havaita katsoessa liikkuvan junan tai auton ikkunasta ulos liikkumissuuntaan nähden kohtisuorassa.

Liikeparallaksiefektin tuottaminen tietokoneen näytölle vaatii periaatteessa ainoastaan katselijan pään sijainnin riittävän tarkkaa laskemista reaaliajassa. Erityisiä näyttöjä tai silmälaseja ei siis tarvita. Pään sijainnin laskemiseen tarvitaan ainoastaan kamera. Joissakin järjestelmissä voidaan myös laskea molempien silmien sijainti ja katsevektori eli suunta, johon henkilö katsoo, mutta liikeparallaksiefektin tuottamiseen riittää yhden pisteen laskeminen. 3D-näkymän perspektiivi muutetaan havaintopisteen mukaan, jolloin perspektiivi ikään kuin seuraa katsojaa ja tästä syntyvä liikeparallaksi-ilmiö tuottaa syvyysvaikutelman. Sivustakatsojalle perspektiivi näyttää tietenkin väärältä, joten yhtä katsojaa kohti tarvitaan yksi näyttöruutu. Ruudut voivat tosin olla samassa näyttölaitteessa esimerkiksi eri ikkunoissa, mutta silti tämä rajoite rajaa liikeparallaksin sovellusalueita.

Suurimpia perinteisten stereoskopiaan perustuvien näyttöjen haittoja on katselukulman kapeus ja tarve silmälasien käyttämiselle. Katselijoiden täytyy myös pysyä melko liikkumatta etenkin autostereoskooppisia näyttöjä katsottaessa. Liikeparallaksilla näitä ongelmia ei ole; se itse asiassa vaatii katselijaa liikkumaan hieman. Liikkeen ei kuitenkaan tarvitse olla kovin suurta. Liikeparallaksiin perustuvan 3D-näytön käyttäminen on myös koettu miellyttävämmäksi [Li et al., 2012], koska käyttäjien ei tarvitse pukea ylimääräisiä laitteita päälle. Liikeparallaksi mahdollistaa myös pidempien syvyysetäisyyksien esittämisen kuin stereoskopia [Uehira et al., 2008].

Sovelluskehittäjien kannalta liikeparallaksin toteuttaminen on melko helppoa. Tarvittavat laitteistot saa hankittua halvalla ja valmiita ohjelmistokomponentteja käyttäjän pään seuraamiseen, kuten OpenCV, on saatavilla avoimena lähdekoodina. Käytännössä kaikki 3D-rajapinnat mahdollistavat perspektiivin laskemisen, ja sen toteuttaminen on helppoa vaikka oikean persketiivin laskeminen ei olekaan triviaalia. Tarkkaa toteutusta käsitellään myöhemmin.

Vaikka toistaiseksi liikeparallaksiin perustuvia 3D-sovelluksia ei ole juuri ollenkaan kuluttajamarkkinoilla, on sitä tutkittu ja siitä on kehitetty prototyyppijärjestelmiä.

Suenaga ja muut [2005] esittelevät liikeparallaksiin perustuvaa 3D-näyttöä. Heidän kehittämässä järjestelmässä pään sijainnin lisäksi lasketaan myös kolmiulotteinen katsevektori. Järjestelmässä käytetään kahta RGB-kameraa, jotka on kiinnitetty moottoroituun alustaan, jolloin ne voivat seurata katselijan liikkeitä. Resoluutiolla 320 kertaa 240 kuvapistettä virhemarginaali pään sijainnissa oli 2 mm ja katseluvektorissa 5 astetta ja järjestelmän prosessointinopeus pysyi 30 fps:n sisällä. Suenaga ja muut raportoivat myös tutkimuksesta, jonka mukaan käyttäjä pystyi havaitsemaan syvyyden esitetyllä järjestelmällä.

3. Liikeparallaksin toteutus

Liikeparallaksiefektin tuottaminen vaatii käyttäjän katselupisteen laskemisen ja 3D-näkymän perspektiivin muuttamisen katselupisteen mukaan reaaliajassa. Käyttäjän katselupisteeksi riittää yksi piste käyttäjän pään alueelta.

Käyttäjän pään seuraamiseen voidaan käyttää erilaisia menetelmiä ja erilaisia laitteita. Näistä annetaan esimerkkejä myöhemmin. Periaatteessa tavallinen videokamera tai webkamera riittää käyttäjän seuraamiseen, mutta paremman tuloksen saa syvyyskameralla, joita on viime vuosina tullut kuluttajamarkkinoille ja joihin on julkaistu kaupallisia ja avoimen lähdekoodin rajapintoja sovelluskehittäjille.

Liikeparallaksin keskeisin kysymys on oikean perspektiivin laskeminen 3D-näkymälle. Sen laskeminen ei ole triviaalia, eikä tällä hetkellä tarvittavia toiminnallisuuksia ole tarjota valmiissa rajapinnoissa tai ohjelmistokomponenteissa.

Modernit 3D-rajapinnat, joista tunnetuimmat ovat OpenGL ja Direct3D, käyttävät projektiomatriisia laskemaan kolmiulotteisille objekteille kaksiulotteiset ruutukoordinaatit [Watt, 2001]. Molemmat rajapinnat tarjoavat ortogonaali- ja perspektiiviprojektion. Perspektiiviprojektio tuottaa aidomman kolmiulotteisen näkymän, sillä sen ansiosta kauempana olevat objektit näyttävät pienemmiltä ruudulla, ja on siten käytetympi. Perspektiiviprojektion tarkka toteutus vaihtelee eri rajapinnoilla, mutta yleensä se esitetään matriisina kuvassa 2 esitetyssä muodossa.

$$P = \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-f+n}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

Kuva 2. Perspektiiviprojektiomatriisi

Kuvan 2 projektiomatriisi esittää perspektiivin yleisessä muodossa. Toistaiseksi 3D-rajapintojen tarjoamat apufunktiot projektion laskemiseen kuitenkin tekevät tiettyjä oletuksia, joten tarvittavat arvot on laskettava sovellustasolla ja syötettävä suoraan matriisiin. Tähän rajapinnat antavat tarvittavat funktiot. Esimerkiksi OpenGL-rajapinnassa vastaava funktio on glFrustum(left, right, bottom, top, near, far), jossa parametri *left* vastaa matriisin arvoa *l*, *right* vastaa arvoa *r*, *bottom* vastaa arvoa *b*, *top* vastaa arvoa *t*, *near* vastaa arvoa *n* ja *far* vastaa arvoa *f* kuvan 2 matriisissa.

Perspektiiviprojektion laskemisessa on otettava huomioon, että katselupiste ei pysy kohtisuorassa ruudun keskikohtaan nähden. Katselupisteen ollessa kohtisuorassa projektio muodostaa tasasivuisen pyramidin, jolloin arvojen *l* ja *r* itseisarvot ja arvojen *t* ja *b* itseisarvot ovat yhtä suuret. Tämä on oletustilanne 3D-rajapintojen apufunktioissa. Liikeparallaksin kannalta tämä on vain yksi erityistilanne ja nämä oletukset on kierrettävä.

Yllä esitetyt arvot *left*, *right*, *top* ja *bottom* lasketaan katselupisteen ruututasolle projisoidusta pisteestä p_{pe}. Nämä arvot lasketaan erotuksena pisteestä p_{pe} ruututason reunoille, joten ruudun reunojen koordinaatit on myös tunnettava. Arvot lasketaan ruututason muodostamassa avaruudessa, jonka

origoksi määritellään seurantalaitteen sijainti samassa avaruudessa. Periaatteessa tähän käy muukin piste, mutta ylimääräisten muunnosten välttämiseksi on syytä valita se piste, josta lasketaan käyttäjän pään sijainti. Näiden arvojen lisäksi projektiomatriisiin syötetään leikkaustason *far-* ja *near*-arvot, jotka voidaan valita melko vapaasti.

Kun seuraaminen tapahtuu reaaliajassa ja perspektiiviprojektio lasketaan jokaiselle kuvalle, syntyy sulava liikeparallaksi-ilmiö. Ruudun päivittämisen tulisi pysyä 30 fps:n eli 30 Hz:in yläpuolella. Tätä rajaa pidetään usein 3D-grafiikassa hyväksyttävänä rajana, jonka alle menevät päivitysnopeudet ihminen pystyy havaitsemaan ja liikkeen sulavuus kärsii.

Seuraavaksi tarkastellaan, millaisilla laitteilla liikeparallaksi-ilmiö voidaan tuottaa tavallisille ruuduille.

3.1. Laitteistot

Liikeparallaksin suurimpia etuja on se, että se ei vaadi katselijan ylle puettavia ylimääräisiä laitteita, vaan se voidaan toteuttaa käyttämällä tavallista kameraa, jollaisia löytyy nykyisin lähes jokaisesta tietokoneesta. Joissakin tutkimuskohteina olleissa järjestelmissä on kuitenkin käytetty ylimääräisiä puettavia laitteita paremman tarkkuuden tai käytetyn teknologian rajoitteiden takia.

Tavallinen RGB-tekniikaan pohjautuva web- tai videokamera on riittävä pään seurantaan. Muun muassa Suenagan ja muiden [2005] kehittämässä järjestelmässä käytetään kahta videokameraa. Kameran tuottamasta videokuvasta etsitään konenäkömenetelmillä ihmishahmoja ja erityisesti kiinnostavia pisteitä, tässä tapauksessa pään keskipiste. Suenagan ja muiden järjestelmässä laskettiin myös molempien silmien sijainti ja vektori katseen suunnalle. Konenäkömenetelmillä saadaan laskettua hahmon sijainti tarpeeksi tehokkaasti ja tarkasti syvyyttä lukuun ottamatta. Koska tavallisen kameran tuottamassa datassa ei ole syvyysarvoja mukana (se on käytännössä kaksiulotteinen), pitää etäisyys laskea jollakin menetelmällä sovellustasolla. Tämä lisää epätarkkuutta.

Tavallista kameraa tehokkaampi vaihtoehto on syvyyskamera kuten, Microsoftin Kinect. Tällaisia laitteita on viime vuosina tullut kuluttajamarkkinoille, ja seuraavan polven tietokoneissa niitä on mahdollisesti valmiiksi integroituna webkameran lisäksi. Syvyyskameran tuottama kuva ei ole tavallinen RGB-värikartta (laitteissa usein on myös tällainen tavallinen kamera), vaan infrapunasäteillä tai jollain muulla tekniikalla rakennettu syvyyskartta. Kartan jokainen kuvapiste esittää etäisyyttä kohteeseen kyseisessä pisteessä.

Infrapunaan perustuvan syvyyskameran toiminta perustuu laitteesta lähetettäviin infrapunasäteisiin, jotka kimpoavat ympäröivistä pinnoista takaisin

syvyyskameraan. Kamerassa oleva infrapunatunnistin laskee takaisin tulleen säteen voimakkuudesta sen kimmottaneen pinnan etäisyyden. Eli mitä kauempana pinta on, josta säde kimposi, sitä heikompi on takaisintuleva säde. Syvyyskameran tuottama kartta siis kertoo suoraan etäisyydet kuvapisteissä, mikä säästää hieman prosessointiaikaa sovellustasolla. Syvyyskartan datasta on kuitenkin etsittävä hahmontunnistusmenetelmillä ihmishahmoja.

Katselijan seuraaminen voidaan toteuttaa myös puettavilla laitteilla, usein ihoon, vaatteisiin tai silmälaseihin kiinnitettävillä tageilla. Tagit käyttävät jotain seurantamenetelmää, esimerkiksi infrapunaa, jota voidaan seurata tarkasti kameralla. Puettavat laitteet tekevät katselijan seuraamisesta tehokkaampaa ja tarkempaa, mutta ne vaikuttavat käyttökokemukseen ja immersioon.

3.2. Ohjelmistot

Kamerasta tulevaa raakadataa on käsiteltävä konenäkömenetelmillä haluttujen hahmojen löytämiseksi. Molemmilla syötteillä (RGB- ja syvyyskartta) menetelmät on samankaltaisia, mutta käytännössä data on niin erilaisessa muodossa, ettei ole mahdollista käyttää yhtä ja samaa algoritmin toteutusta molemmissa tapauksissa, vaikka periaatteeltaan hahmontunnistusalgoritmi olisi samanlainen.

Päänseurantaan tarkoitettuja konenäköalgoritmeja on kehitetty ja tutkittu kattavasti, ja prosessoreiden tehojen kasvaessa uusia menetelmiä kehitetään jatkuvasti [Ryu and Kim, 2007; Toyama, 1999]. Ryu ja Kim [2007] esittävät algoritmin, joka yhdistää eri menetelmiä nopean ja luotettavan pään seurannan saavuttamiseksi. Algoritmin testaamiseen käytetyssä tietokoneessa oli 3,2 GHz Pentium IV -prosessori ja Logiteching webkamera. Järjestelmän kuvanprosessointinopeus testilaitteilla tosin oli 10-15 fps:n välillä, mikä ei ole riittävä sulavan liikeparallaksin tuottamiseksi. Nykyiset prosessorit todennäköisesti tuottaisivat nopeamman kuvanprosessoinnin, mutta tehokkuuden ja luotettavuuden välillä on aina tehtävä kompromissi.

Myös Götürk ja Tomasi [2004] esittelevät päänseuranta-algoritmin, jonka tunnistaminen perustuu korrelaatiopohjaiseen painotettuun interpolaatioon. Aluksi algoritmille syötetään opetusdataa, minkä jälkeen se tunnistaa syötteestä pään sijainnin. Kyseessä on siis ohjattuun oppimiseen perustuva algoritmi.

Kaikkien käytettyjen algoritmien ja laskennan tulisi olla mahdollisimman tehokasta ja optimoitua, sillä ensinnäkin laskenta on monimutkaista (useiden hahmontunnistusalgoritmien aikakompleksisuus on eksponentiaalinen) ja toiseksi samalla prosessorilla ja näytönohjaimella usein pitää laskea myös mahdollisesti monimutkainen 3D-näkymä.

Kuvanpäivityksen tulee tapahtua mahdollisimman nopeasti, jotta liikeparallaksiefektin tuottama illuusio tuntuisi todelliselta. Yuanin ja muiden [2000] tutkimuksen mukaan viive katselijan liikkeen alkamisen ja näkymän muuttumisen välillä saa olla korkeintaan 265 millisekuntia. Tämä viive tuntuu melko pitkältä, ja nykyisillä laitteistolla päästään huomattavasti alhaisempiin viiveisiin.

Monimutkaiset hahmontunnistusalgoritmit ovat vaikeita ja resursseja vaativia kehittää, joten sovelluskehittäjille ei usein ole realistista tai mahdollista toteuttaa koko järjestelmää alusta alkaen. Syvyyskameroiden yleistyessä useammat osapuolet ovatkin kehittäneet valmiita kirjastoja ja rajapintoja sovelluskehittäjille. Microsoft tarjoaa omaan .NET-ympäristöön kuuluvan kirjaston syvyyskameran tuottaman datan käsittelemiseen. Vaihtoehtona on myös PrimeSensen kehittämä avoimen lähdekoodin ohjelmistokehyksen OpenNI:n.

3.3. Innosummer-projekti *3D experience on 2D display*

Kesällä 2011 kehitimme liikeparallaksiin perustuvan esittelysovelluksen Demolan järjestämässä *InnoSummer*-kesätyöprojektissa. Projektimme oli nimeltään *3D on 2D.* Tuotoksemme on yksinkertainen akvaario-tyylinen sovellus eli sen funktio on ainoastaan antaa käyttäjälle jotain katseltavaa ja tässä tapauksessa tietenkin esitellä liikeparallaksiefektiä.

Sovellus esittää käyttäjälle ikkunan avaruuteen, jossa näkyy tähtiä, planeettoja ja muita taivaankappaleita eri etäisyyksillä. Näkymän edustalla myös on avaruusaluksen kansi ja apulaisrobotti. Kannen ja robotin z-koordinaatit ovat osittain näyttötason edessä, joten ne näyttävät tulevan hieman ulos näytöstä.

Liikeparallaksin lisäksi kehitimme sovellukseen kokeilumielessä yksinkertaisen eleohjauksen. Kun käyttäjä osoittaa oikealla kädellään kohti näkymässä olevaa planeettaa, se valitaan ja valinnan merkiksi planeetan tekstuuria korostetaan ja soitetaan äänimerkki. Kun planeetta on valittuna ja käyttäjä nostaa vasemman käden ylös, tulee näkyviin infopaneeli, jossa lukee tietoa planeetasta. Infopaneeli on kaksiulotteinen taso, jonka z-koordinaatit ovat selkeästi näyttötason edessä. Paneeli näyttää leijuvan katselijan ja näytön välissä. Pyyhkäisemällä oikealla kädellä oikealta vasemmalla infopaneelin saa poistettua näkyviltä.

Kehitimme sovelluksen Microsoftin Kinect -syvyyskameralaitteella. Rajapintana laitteelle käytimme avoimeen lähdekoodiin perustuvaa OpenNI-ohjelmistokehystä (http://openni.org/). OpenNI on suunniteltu tarjoamaan yleisen ja laiteriippumattoman rajapinnan ohjelmistokehittäjille, joten sovelluksemme toimii myös muilla syvyyskameroilla. Sovellus on testattu myös Asusksen Xtion-

syvyyskameralaitteella. Sovelluksemme 3D-moottorina käytimme myös avoimeen lähdekoodiin perustuvaa Ogre3D-grafiikkakehystä.

Ogre3D:lle on kehitetty erilaisia komponentteja, jotka toimivat suoraan OpenNI:n kanssa tarjoten erilaisia toiminnallisuuksia, mutta ainakaan projektimme kehittämisen aikana ei ollut valmista komponenttia liikeparallaksille. Sen toteutimme itse.

Sovelluksemme käyttää OpenNI:n tuottamaa katselijan sijainnin arvoja ja syöttää ne Ogre3D:n perspektiivimatriisiin. Vaikka Ogre3D on tarkoitettu hieman korkeamman tason grafiikkakehykseksi, se tarjoaa myös mahdollisuuden manipuloida suoraan projektiomatriisia funktiolla Ogre::Camera::setFrustumExtents(left, right, top, bottom), kun parametrit left, right, top ja bottom ovat liukuluja ja esittävät aikaisemmin mainitut etäisyydet päänsijainnin projisoidusta pisteestä näyttötasolla sen reunoille.

4. Liikeparallaksin sovelluksia

Liikeparallaksilla ja stereoskopialla saadaan siis aikaan realistisen tuntuisia kolmiulotteisia havaintoja. Menetelmien toimintaperiaatteet kuitenkin eroavat toisistaan kuten aiemmin on esitetty, ja tällä on vaikutusta minkälaisiin sovelluksiin menetelmiä voidaan hyödyntää. Syvyyden hahmottamista määrittelevissä tutkimuksissa on tullut esille, että liikeparallaksin ja stereoskopian tuottamat syvyysvaikutelmat eivät ole täysin samanlaiset ja valitulla menetelmällä on vaikutusta siihen miten katselija hahmottaa kolmiulotteisen näkymän [Jones et al., 2008; Naeplfin and Menozzi, 2001]. Nämä erot eivät liity yksiselitteisesti etäisyyksien arviointiin, vaan kyseessä on monimutkaisempi ilmiö. 3D-sovellusten kehittämisessä on siis erityisen tärkeää kiinnittää huomiota siihen, millä menetelmällä saadaan paras vaikutelma kyseiseen sovellukseen.

Yhdessä sovelluksessa voidaan tietenkin käyttää useita menetelmiä. Cipiloglu ja muut [2010] esittelevät järjestelmän, joka reaaliajassa määrittelee jokaiselle tilanteelle parhaan ja tehokkaimman tavan syvyysvaikutelman tuottamiseen. Heidän menetelmänsä perustuu sumeaan logiikkaan ja se ottaa huomioon jokaiseen näkymään liittyviä parametreja, kuten näkymän asetteluun liittyviä ja sen tehtävään liittyviä. Cipiloglu ja muut [2010] esittelevät myös tuloksia kokeesta, jolla arvioitiin heidän menetelmää käytännössä. Tulosten mukaan syvyyden arvioinnissa oli vain 3,1% virhe ja verrattuna muihin testitapauksiin heidän menetelmänsä sai parhaat pisteet.

Käyttäjän kannalta suurin ero liikeparallaksin ja stereoskopian välillä on siinä, miten käyttäjän oletetaan käyttäytyvän sovellusta käytettäessä. Liikeparallaksi vaatii käyttäjää liikkumaan hieman, mutta tämä ei tarkoita, että käyt-

täjän pitäisi tehdä suuria liikkeitä seisten. Jo muutaman senttimetrin pään liike riittää tuottamaan liikeparallaksi-ilmiön ja harva käyttäjä istuu tietokoneen tai muun päätteen edessä täysin liikkumatta. Liikkeen suuruus täytyy kuitenkin ottaa huomioon sovelluksen kehittämisessä. Jos näkymässä olevat objektit ovat suhteessa kaukana, tarvitaan myös suurempaa liikettä parallaksin havaitsemiseksi ja päin vastoin.

Sovelluskehittäjien kannalta liikeparallaksin käyttöönotto aikaisemmin kehitetyssä sovelluksessa ei vaadi suuria muutoksia ohjelman lähdekoodiin eikä mitään muutoksia sisältöön eli 3D-malleihin ja dataan. Datan muuntaminen uuteen muotoon on usein hyvin suuri investointi. Stereoskopian käyttäminen puolestaan saattaa vaatia myös sisällön konvertoimista ja laajempia muutoksia lähdekoodiin.

Seuraavaksi pohditaan, miten liikeparallaksi ja stereoskopia soveltuvat neljään sovellusalueeseen: käyttöliittymiin, jotka ovat perustana muille sovellusalueille, peleihin ja viihdesovelluksiin, teollisuuden sovelluksiin ja lisätty- ja virtuaalitodellisuussovelluksiin. Sovellusalueet liittyvät monin tavoin läheisesti toisiinsa ja ovat osittain päällekkäisiä, mutta sovelluskehityksen näkökulmasta ja kirjallisuuskatsauksen perusteella niitä on syytä käsitellä edellä mainituissa osissa.

4.1. 3D-käyttöliittymät

3D-käyttöliittymät ovat yleistyneet edellisten vuosien aikana ja ne todennäköisesti jatkavat yleistymistään [Bowman et al., 2008; Cervantes et al., 2012].

Bowman ja muut arvioivat useita eri teknologioita, joilla voidaan ohjata kolmiulotteista käyttöliittymää. Näitä menetelmiä voidaan myös soveltaa liikeparallaksiin perustuvassa 3D-sovelluksissa. Bowman ja muut ovat myös tutkineet 3D-käyttöliittymien käytettävyyttä etenkin uuden käyttäjän näkökulmasta, ja he ovat laatineet kahdeksan periaatetta 3D-käyttöliittymien suunnitteluun.

3D-käyttöliittymät ovat kuitenkin vielä harvinaisia. Leen ja Greenen [2005] mukaan yksi syy tähän on standardien puuttuminen ja laitekokoonpanojen monimuotoisuus. He esittelevät ohjelmistokehyksen, joka laajentaa Grapplnimisen ohjelmistokehyksen toiminnallisuuksia. Leen ja Greenen ohjelmistokehys on tarkoitettu automatisoimaan 3D-käyttöliittymien rakentamista ohjelman ajonaikana perustuen saatavilla oleviin laitteisiin ja rajapintoihin. Sen avulla ohjelmistokehittäjien ei tarvitse huolehtia useista eri kokoonpanoista, vaan he voivat keskittyä määrittelemään käyttöliittymään kuuluvia elementtejä.

Parallaksiefekti tuo uusia näkökulmia käyttöliittymien kehittämiseen. Koska käyttäjä voi liikkua ja katsella näkymää useasta kulmasta, luonnollista olisi, että käyttäjä ei enää istuisi näytön edessä. Se periaatteessa toimii myös perinteisissä hiiri ja näppäimistö -yhdistelmässä, mutta luonnoista olisi toteuttaa elekäyttöliittymä. Usealla liikeparallaksin toteutusmenetelmällä voidaan suoraan seurata käyttäjän muita raajoa, ja elekäyttöliittymä saadaan samalla kertaa.

Toinen mahdollinen käyttöliittymien sovellusmuoto liikeparallaksille yhdistettynä elekäyttöliittymään on julkisissa tiloissa olevat käyttöliittymät [Uehira et al., 2008]. Nykyisin julkisissa käyttöliittymissä käytetään usein kosketusnäyttöjä, jotka voivat joillekin henkilöille olla epämiellyttäviä esimerkiksi hygieniasyistä. Tämä sopisi esimerkiksi mainostamiseen.

Valtaosa Bowmanin ja muiden tutkimista käyttöliittymistä toimivat suurempien eleiden kautta. Pelkällä stereoskoppisella menetelmällä tämä voi johtaa ongelmiin käytettävyydessä, sillä kuten aiemmin on esitetty stereoskooppinen tekniikka vaatti usein käyttäjää olemaan melko liikkumatta ja perspektiivin vaihtumisen puuttuminen voi pilata 3D-vaikutelman.

Liikeparallaksiin perustuva 3D-käyttöliittymä voisi myös parantaa ergonomista työskentelyä. Useiden tutkimusten mukaan liiallinen istuminen on haitaksi terveydelle. Seisoen käytettävä eleohjauskäyttöliittymä myös lisää liikkumista.

4.2. Pelit ja viihdesovellukset

Tietokonepelit edustavat usein uusinta teknologian kärkeä erityisesti grafiikan ja käyttöliittymän osalta. 3D-näytöt eivät ole tässäkään poikkeus. Peliteollisuus on jo yli kolmen vuosikymmenen ajan jatkuvasti hakenut uusia käyttöliittymiä uusien pelaajien houkuttelemiseksi ja seuraavaksi näyttää olevan vuorossa 3D-näytöt ja -käyttöliittymät [LaViola, 2008].

Liikeparallaksia ei tällä hetkellä käytetä tietokonepeleissä juuri ollenkaan, mutta juuri peleissä sillä saadaan lisäarvoa. Viime vuosien aikana markkinoille tulleet uudet liikkeeseen perustuvat peliohjainlaitteet, kuten Nintendo Wii, Microsoft Kinect ja Sony Move ovat tuoneet 3D-käyttöliittymät konsolipeleihin. Niiden kanssa pelaajat joutuvat usein liikkumaan, jolloin näkymän perspektiivin vaihtuminen olisi luonnollista ja sillä saataisiin kehitettyä uusia pelimekanismeja [Li et al., 2012]. Liikeparallaksi voidaan lisätä myös tietokonepeleihin, vaikka usein PC-pelejä pelataan hiirellä ja näppäimistöllä. Pienelläkin pään liikkeellä on vaikutus perspektiiviin, etenkin jos näyttö on suhteellisen lähellä katselijaa ja pelin 3D-näkymä on suunniteltu oikein.

Liikeparallaksia voidaan soveltaa peleissä laajemminkin. Vaikka se vaatii vähintään yhden ruudun katselijaa kohti, usea pelaaja voi käyttää samaa näyttölaitetta jaetulla ruudulla. Jokaiselle katselijalle siis jaetaan alue yhdeltä näytöltä. Tätä periaatetta on käytetty tietokonepeleissä laajasti. Ohjelmallisesti jokaisen alueen perspektiiviprojektio voidaan laskea erikseen. Liikeparallaksi voi siten myös toimia sosiaalisissa peleissä ja muissa sovelluksissa.

3D-näkymiä voidaan käyttää myös erilaisissa viihdesovelluksissa. Nykyisin varmasti tunnetuin näistä on 3D-elokuvat. Elokuvateattereissa esitettävät 3D-elokuvat perustuvat poikkeuksetta stereoskopiaan. Parallaksiefekti tarvitsee aina yhden näyttölaitteen yhtä katselijaa kohti, joten elokuvateattereissa siitä ei saada hyötyä. Yksin katsottaessa parallaksiefektiä voisi ehkä harkita elokuviin, mutta parallaksi vaatii katselijan liikkumista, mikä ei välttämättä ole toivottavaa elokuvaa katsellessa. Tosin parallaksin vaatima liike on tietyissä tilanteissa hyvin pieni.

Toinen viihdesovellusalue on erilaiset virtuaaliset maisemaikkunat ja niin sanotut akvaariosovellukset. Psykologisissa tutkimuksissa on huomattu, että ikkunattomissa huoneistoissa eläminen ja työskenteleminen rasittavat ihmisten psyykettä [Radikovic et al., 2005]. Parallaksiefektillä voidaan luoda keinotekoisia ikkunoita joko näyttölaitteelle tai videoprojektorilla heijastettuna seinälle. Näissä sovelluksissa on tärkeää, että 3D-illuusio on todentuntuinen ja että perspektiivi vaihtuu kun katsoja tarkastelee näkymää eri kulmista. Jatkuva liikeparallaksi tuottaa tässä tapauksessa todentuntuisemman ja hyödyllisemmän vaikutelman myös siksi, että käyttäjän ei tarvitse pukea ylimääräisiä laitteita.

Radirovik ja muut [2005] ovat kehittäneet tällaisen keinotekoisen ikkunan. Heidän tutkimuksensa mukaan parallaksiefektillä toteutettu ikkuna tarjoaa helpotusta ikkunattomissa huoneistoissa eläville ja työskenteleville henkilöille. Myös projektimme *3D on 2D* kuuluu myös tähän kategoriaan.

Samaa ajattelua seuraten voidaan kuvitella, että liikeparallaksiin pohjaten voidaan tehdä myös kolmiulotteisia ja mahdollisesti interaktiivisia digitaalisia taideteoksia. Ogi ja muut [2012] ovatkin kehittäneet digitaalisen version perinteisestä japanilaisesta *ukiyoe*-puupiirroksesta, joka myös hyödyntää liikeparallaksia tuomaan syvyysvaikutelman kuvaan.

4.3. Teollisuus ja lääketiede

Useilla eri teollisuuden aloilla tehdään kolmiulotteisia malleja ja mallintamista reaalimaailman objekteista, kuten rakennusarkkitehtuurissa, tuotantoteollisuudessa ja peli- ja viihdeteollisuudessa. Kaikilla näillä aloilla on mahdollista saada hyötyä 3D-näytöistä mallien valmistamisessa ja niiden esittelyssä.

Parallaksiefektiin perustuen voidaan kehittää esittelytyökaluja eri alojen mallintajille. Esimerkiksi rakennusarkkitehdit voisivat hyötyä sovelluksesta, jolla voidaan esittää asiakkaille kolmiulotteista mallia rakennuksesta.

Myös lääketieteestä löytyy monia sovelluskohteita kolmiulotteisille näytöille [Lu et al., 2007; Gallo et al., 2010]. Modernit kehon kuvantamismenetelmät, kuten funktionaalinen magneettiresonanssikuvaus ja tietokonekerroskuvaus, tuottavat kuvattavasta elimestä kaksiulotteisia poikkileikkauksia, joista voidaan rakentaa kolmiulotteinen malli [Gallo et al., 2010]. Gallo ja muut esittelevät lääketieteellisen 3D-datan esittämiseen tarkoitetun järjestelmän. Esitelty järjestelmä käyttää stereoskopiaa syvyyshavainnon perustana, mutta vastaava järjestelmä voitaisiin kehittää myös liikeparallaksiin pohjaten. Lääketieteen sovelluksissa on erityisen tärkeää esitetyn mallin hahmottaminen tarkkojen etäisyyksien arvioimisen sijaan [Gallo et al., 2010]. Gallon ja muiden järjestelmään lisäksi kehitettiin käyttöliittymä mallien manipulointiin. Vastaavanlainen käyttöliittymä voidaan kehittää myös eleohjauksella.

4.4. Lisätty- ja virtuaalitodellisuusympäristöt

Kaikkia kolmiulotteisen havainnon tuottavia sovelluksia voidaan periaatteessa pitää jonkinlaisina lisätty- tai virtuaalitodellisuussovelluksina [LaViola, 2008; Jones et al., 2008]. Toisaalta lisättytodellisuudella usein tarkoitetaan järjestelmää, joka tuo virtuaalisia elementtejä todellisten reaalimaailman elementtien lisäksi ja jotka tavalla tai toisella ovat vuorovaikutuksessa todellisen ympäristön kanssa. Vuorovaikutus voi myös olla passiivista, mutta usein jonkinlainen yhteys säilyy elementtien välillä. Lisäksi kaikki lisättytodellisuussovellukset eivät välttämättä tarvitse syvyyshavaintoa ollenkaan. Tästä näkökulmasta katsottuna edellä esitellyt sovellusalueet eivät useassa tapauksessa kuulu lisättytodellisuuden piiriin. Virtuaalitodellisuudella puolestaan tarkoitetaan kokonaan tietokoneella generoitua ympäristöä. Tässä tutkielmassa otetaan tämä näkökulma ja lisätty- ja virtuaalitodellisuussovelluksia käsitellään erillisenä sovellusalueena.

Jones ja muut [2008] tutkivat liikeparallaksin ja stereoskopian ja näiden yhteisvaikutusta syvyyshavaintoon lisätty- ja virtuaalitodellisuusympäristöissä. He keskittyivät tutkimaan etäisyyksien arviointia. Tulosten mukaan etäisyyksien arvioinnissa tapahtui aliarviointia virtuaalitodellisuudessa, mutta ei lisättytodellisuudessa. Lisäksi tutkimuksessa huomattiin, että liikeparallaksi itse asiassa lisäsi virhettä etäisyyksien arvioinnissa. Tämän syyksi Jones ja muut epäilevät kokeissa käytetyn päähän laitettavan laitteen painoa. Ilman puettavia laitteita liikeparallaksilla voisi mahdollisesti tuottaa tarkemman syvyyshavain-

non. Liikeparallaksi siis useissa tapauksissa soveltuu paremmin lisättytodellisuusympäristöihin kuin virtuaalitodellisuuteen.

Myös Mulder ja muut [2003] esittelevät molempiin liikeparallaksiin ja stereoskopiaan perustuvan lisätty- ja virtuaalitodellisuussovelluksen. Heidän järjestelmänsä koostuu sulkijalaseista, laseihin kiinnitettävästä seurantamerkistä, kahdesta webkamerasta, näytöstä ja puoliläpäisevästä peilistä, jolla voidaan simuloida esineiden manipulointia lisättytodellisuusympäristössä. Mulderin ja muiden järjestelmä toimii tehokkaasti ja tarkasti, mutta käytettävyyden kannalta järjestelmä sitoo käyttäjää liikaa soveltuakseen sulavaa käyttöä vaativiin tilanteisiin.

Tässä tutkielmassa esitettyä tapaa tuottaa liikeparallaksi-ilmiö voidaan soveltaa sellaisenaan myös lisätty- ja virtuaalitodellisuuteen, mutta usein näissä tarvitaan myös muita tekniikoita. Lisättytodellisuudessa usein tarvitsee myös jollain tavalla mallintaa tai hahmottaa ympäristöä. Virtuaalitodellisuudessa taas usein halutaan täysin virtuaalisesti tuotettu ympäristö. Liikeparallaksi soveltuu myös tähän tarkoitukseen esimerkiksi heijastamalla yhdellä tai useammalla videoprojektorilla 3D-näkymä huoneen jokaiselle seinälle. Tässäkin tapauksessa riittää, että tiedetään katselijan sijainti ja katseen suunta huoneessa, mutta jokaiselle näkymälle täytyy laskea oma perspektiiviprojektio, mikä on monimutkaisempi kuin aiemmin esitelty.

Eräs mielenkiintoinen ja toistaiseksi teoreettinen lisättytodellisuussovelluskohde löytyy mobiililaitteden puolelta. Mobiililaitteiden laitteiston laskentateho alkaa olla jo tarpeeksi tehokas hahmontunnistuksen ja muun vaativan laskennan suorittamiseen. Useissa älypuhelimissa ja tablet-laitteissa on kamera sekä edessä että takana. Etukameraa voitaisiin käyttää käyttäjän katseen seuraamiseen liikeparallaksin tuottamiseksi ja takakameraa ympäristön skannaamiseen. Skannatusta ympäristöstä voidaan rakentaa 3D-malli, johon voidaan sijoittaa virtuaalisia objekteja. Tällä menetelmällä voitaisiin kehittää hyvin mielenkiintoisia sovelluksia, kuten pelejä ja navigointisovelluksia, mutta sitä ennen menetelmää on tutkittava laajasti.

5. Yhteenveto

Tässä tutkielmassa esitettiin, miten liikeparallaksia voidaan hyödyntää 3D-näyttöjen ja -sovellusten kehittämisessä. Stereoskoopiaan, joka on vielä nykyisin käytetympi menetelmä 3D-näytöissä, verrattuna liikeparallaksilla saavutetaan sulavampi ja miellyttävämpi käyttökokemus. Aikaisemmat tutkimustulokset stereoskopian ja liikeparallaksin tuottamista syvyysvaikutelmista kuitenkin osoittavat, että stereoskopialla saavutetaan tarkempi etäisyyksien havaitsemi-

nen, kun taas liikeparallaksilla näkymän hahmottaminen on selkeämpää. Molempien yhdistelmällä saavutetaan paras lopputulos.

Liikeparallaksin toteuttaminen vaatii oikean perspektiiviprojektion laskemisen reaaliajassa käyttäjän katselupisteen mukaan. Käyttäjän katselupisteen määrittelemiseksi riittää tavallinen kamera, mutta syvyyskameralla saadaan tehostettua prosessointia. Sovelluskehittäjien kannalta liikeparallaksin toteuttaminen on monissa tapauksissa helpompaa ja halvempaa kuin stereoskopian.

Molempia syvyyshavainnon tuottavia menetelmiä voidaan käyttää useissa sovellusalueissa. Liikeparallaksia hyödyntämällä voidaan kuitenkin kehittää uudenlaisia sovelluksia, kuten käyttöliittymiä, pelejä, teollisuuden ja lääketieteen sovelluksia ja lisätty- ja virtuaalitodellisuussovelluksia.

Liikeparallaksia voidaan mahdollisesti tulevaisuudessa soveltaa myös mobiililaitteilla, mikä avaa paljon uusia ja mielenkiintoisia mahdollisuuksia sovelluskehittäjille.

Viiteluettelo

[Bowman *et al.,* 2008] Doug A. Bowman, Sabine Coquillart, Bernd Froelich, Michitaka Hirose, Yosifumi Kitamura, Kiyoshi Kiyokawa and Wolfgang Stuerzlinger, 3D user interfaces: new directions and perspectives, *Computer Graphics and Applications*, **28** 6 (2008), 20-36.

[Cervantes *et al.,* 2012] Jaime Chapinal Cervantes, Francisco Luis Gutiérrez Vela and Patricia Paderewski Rodríguez, Natural interaction using Kinect, In: *Proc. of the 13th International Conference on Interacción Persona-Ordenador*, 2012, **14.**

[Cipiloglu *et al.,* 2010] Zeynep Cipiloglu, Abdullah Bulbul and Tolga Capin, A framework for enhancing depth perception in computer graphics, In: *Proc. of the 7th Symposium on Applied Perception in Graphics and Visualization (APGV '10)*, 2010, 141-148.

[Gallo *et al.,* 2010] L. Gallo, A. Minutolo and G. De Pietro, A user interface for VR-ready 3D medical imaging by off-the-shelf input devices, *Computers in Biology and Medicine*, **40**, 3 (March 2010), 350-358.

[Götürk and Tomasi, 2004] Salih Burak Götürk and Carlo Tomasi, 3D head tracking based on recognition and interpolation using a time-of-flight depth sensor, In: *Proc. of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '04)*, 2004, 211-217.

[Jones *et al.*, 2008] J. Adam Jones, J. Edward Swan II, Gurjot Singh, Eric Kolstad and Stephen R. Ellis, The effects of virtual reality, augmented reality, and motion parallax on egocentric depth perception, In: *Proc. of the 5th*

Symposium on Applied Perception in Graphics and Visualization (APGV '08), 2008, 9-14.

[LaViola, 2008] Jospeh J. LaViola, Bringin VR and spatial 3D interaction to the masses through video games, *Computer Graphics and Applications, IEEE*, **28** 5 (Sept. - Oct. 2008), 10-15.

[Lee and Green, 2005] Wai Leng Lee and Mark Green, A layout framework for 3D user interfances, In: *Proc. of the ACM Symposium on Virtual Reality Software and Technology (VRTS '05)*, 2005, 96-105.

[Li *et al.*, 2012] Ivan K. Y. Li, Edward M. Peek, Burkhard C. Wünsche and Christof Lutteroth, Enhancing 3D applications using stereoscopic 3D and motion Parallax, In: *Proc. of Australasian User Interface Conference (AUIC '12)*, 2012, 59-68

[Lu *et al.*, 2007] Lina Lu, Chunxiao Chen and Wenlian Cheng, Medical imagevisualization using true 3D display technology, In: *Proc. of International Conference on Complex Medical Engineering (CME '07)*, 2007, 914-918.

[Mulder *et al.*, 2003] Jurriaan D. Mulder, Jack Jansen and Arjen van Rhijn, An affordable optical head tracking system for desktop VR/AR systems, In: *Proc. of the Workshop on Virtual Environments (EGVE '03)*, 2003, 215-223

[Naepflin and Menozzi, 2001] U. Naepflin and M. Menozzi, Can movement parallax compensate lacking sterepsis in spatial explorative search tasks?, *Displays*, **22** 5 (Nov. 2001), 157-164.

[Ogi *et al*., 2012] Tetsuro Ogi, Yoshisuke Tateyama, Hao Lu and Erika Ikeda, Digital 3D ukiyoe using the effect of motion Parallaxp, In: *Proc. of 15th International Conference on Network-Based Information Systems (NBiS '12)*, 2012, 534-539.

[Radikovic *et al.*, 2005] Adrijan S. Radikovic, John J. Leggett, John Keyser, Roger S. Ulrich, Artificial window view of nature, In: *Proc. of Extended Abstracts on Human Factors in Computing Systems (CHI EA '05)*, 2005, 1993-1996.

[Rerábek and Ebrahimi, 2012] Rerábek, M. and Ebrahimi, T., Comparison of 3D portable display restitution techniques on stereo and parallax, In: *Proc. of Fourth International Workshop on Quality of Multimedia Experience (QoMEX '12)*, 2012, 80-85.

[Ryu and Kim, 2007] Real-time 3D head tracking and head gesture recognition, In: *Proc. of the 16th IEEE International Symposium on Robot and Human interactive Communication (RO-MAN '07)*, 2007, 169-172.

[Suenaga *et al.*, 2005] Tsuyoshi Suenaga, Yoshio Matsumoto and Tsukasa Ogasawara, 3D display based on motion parallax using non-contact 3D

measurement of head position, In: *Proc. of the 17th Australia Conference on Computer-Human Interaction (OZCHI '05)*, 2005, 1-4.

[Toyama, 1999] Kentaro Toyama, Head parallax tracking for control of a virtual space: a comparison of algorithms, In: *Proc. of IEEE International Conference on Systems, Man, and Cybernetics (SMC '99)*, **6**, 1-6.

[Uehira *et al*., 2008] K. Uehira, M. Suzuki and Y. Kobayashi, Poster: 3-D display using motion parallax for outdoor user interface, In: *Proc. of IEEE Symposium on 3D User Interfaces (3DUI 2008)*, 2008, 165-166.

[Urey *et al.,* 2011] Hakan Urey, Kishore V. Chellappan, Erdem Erden and Phil Surman, State of the art in stereoscopic and autostereoscopic displays, *Proc. of the IEEE* ***99***, 4 (April 2011), 540-555.

[Watt, 2000] Alan Watt, *3D Computer Graphics 3rd Edition.* Addison-Wesley, 2000

[Yuan *et al*., 2000] Hanfeng Yuan, W. L. Sachtler, Nat Durlach and Barbara Shinn-Cunningham, Effects of time delay on depth perception via head-motion parallax in virtual environment systems, *Presence: Teleoperators And Virtual Environments,* **9**, 6 (2000), 638-647.

Signaalidatan etäkäyttö

Paavo Happonen

Tiivistelmä.
Teollisuudessa koneet ja järjestelmät tuottavat paljon mitattua signaalidataa, jota tallennetaan palvelimelle. Palvelin voi olla kaukana käyttäjästä, jolloin datan lukeminen suoraan kaukaisesta tietokannasta ei ole käytännöllistä. Tässä tutkielmassa tutkitaan tapoja esittää dataa tietoverkon yli nopeasti siten, että se säilyttää visuaalisen ja informatiivisen ulkomuotonsa.

Avainsanat ja -sanonnat: etäkäyttö, aikasarjat, ulotteellinen karsinta
CR-luokat: D.2.11, G.1.2

1. Johdanto

Teollisuudessa koneet ja prosessit tuottavat paljon mitattua signaalidataa, jota tallennetaan palvelimelle analysointia varten. Monissa tilanteissa silmämääräinen analyysi tehdään tästä datasta aikasarjagraafin pohjalta. Koska dataa tallennetaan jatkuvasti, tietokantapalvelimet sijaitsevat yleensä lähellä datalähdettä, ja siten usein kaukana dataa analysoivista käyttäjistä. Kerralla esitettävän datan määrä voi olla huomattava, joten ei ole käytännöllistä siirtää käyttäjälle täyttä esitettävää signaalidataa.

Tämä tutkielma pyrkii kuvaamaan tapoja esittää reaaliaikaisesti suuria datamääriä visuaalisesti identtisessä muodossa minimaalisen kommunikaatioväylän yli. Ensin käydään läpi aiempia olemassa olevia sovelluksia ja niiden ominaisuuksia. Seuraavaksi kuvataan etäkäytön vaatimaa arkkitehtuuria. Sitten tutkitaan datan esikäsittelyä tiedon siirtorajoitteiden ja tärkeyden näkökulmasta. Lopuksi kuvataan datan väliaikaiseen tallennukseen liittyviä käytännön ongelmia ja ratkaisuja.

2. Aiemmat sovellukset

Perinteisesti signaalidatan esitystä on ajateltu staattisena esityksenä, eli suorana paperisen graafitulosteen vastikkeena, joka kuvaa joko esiasetettua aikaväliä tai jatkuvaan tuotetun datan tulostetta.

Päivittäiset ja muut aikaväliotokset ovat yleisiä ja hyödyllisiä yleiskuvan saavuttamiseen. Esimerkiksi viikkoraportista näkee tuotantotehokkuuden kehittymisen tai päiväraportista, miten edellisen päivän asetusmuutokset vaikuttavat.

Tällaisen staattisen kuvan generointi on hyvin helppo prosessi. Se voidaan ajaa palvelimella datan lähellä, ja toimittaa käyttäjälle pelkkä tiivis raportti hitaankin verkon yli. Staattinen kuva on kuitenkin riittämätön, jos halutaan katsoa tarkkoja arvoja tarkalta ajan hetkeltä. Tällainen tarve nousee mm. virhetilanteen tai silmämääräisesti huomatun poikkeaman syyn tutkimisesta. Esimerkiksi jos laitteeseen tulee virhe, täytyy pystyä katsomaan, miten siihen liittyvä signaali piirtyi ennen ja jälkeen tapahtumaa, mitä sen tarkat arvot olivat ja miten muut asiaan liittyvät signaalit piirtyivät, ja mitkä olivat niiden mahdollisten muutosten viiveet suhteessa toisiinsa.

Jotta ongelmatilanteet löytyvät helposti, tarvitaan interaktiivinen sovellus, jolla aikaa voi nopeasti säätää ja lukea arvoja suoraan graafista. Tähän käytetään yleisesti jotain matemaattista ohjelmistoa, kuten MATLAB tai Octave, jotka hallitsevat graafien piirron [Mathworks, 2012; Eaton et al., 2012]. Matemaattisten ohjelmistojen toiminta on suunniteltu toteuttamaan laskennallista tarkoitusta, ja ne joutuvat siten työskentelemään täyden datan kanssa. Käytännössä tämä edellyttää, että data haetaan kokonaisuudessaan paikalliselle työasemalle tai ajetaan ohjelmistoja etätyöpöytäyhteyden yli.

Kun esitettävän datan määrä nousee, sovelluksen käyttö hidastuu, ja siten datan selauksesta tulee hyvin staattinen kokemus. Tällaisissa sovelluksissa ajassa hypitään askelissa ja kosketus aikajanan suhteellisiin tapahtumiin hämärtyy. Aikaisemmin käytetyn piirturitulosteen etu oli se, että sitä pystyi katsomaan tarkalta ajanhetkeltä säilyttäen silti ajallisen asiayhteyden.

3. Asiakas-palvelin -arkkitehtuuri

Koska dataa tuottavat laitteet tai prosessit tuottavat jatkuvasti paljon dataa, on datan tallennukseen yleensä luotu tietokantapalvelin lähelle datalähdettä. Mutta koska käyttäjä usein on hyvinkin kaukana palvelimesta, muodostuu tiedon saamisesta ongelma: datan huomattava määrä tekee suoran tietokantayhteyden käytännössä mahdottomaksi.

Tietokannan siirto läheisempään palvelimeen on myös hidas prosessi ja aina jäljessä todellisesta datasta. Etätyöpöytäyhteys taas kärsii hitaasta käyttöliittymästä, ja hidastuu lisää joutuessaan käsittelemään ja esittämään valtavia datamääriä.

Lopullisesta kuvasta ei kuitenkaan erota kaikkia esitettäviä datapisteitä, joten siirrettävää dataa voi karsia siltä osin, missä se on näkymättömissä. Erotamme siis palvelimen ja asiakkaan karsinta-algoritmilla. Tällöin palvelin ottaa tietokantakutsuun aiemman aika-aluemääritteen lisäksi myös datan tarkkuuden määrityksen ja palauttaa halutulla tarkkuudella minimaalisen määrän da-

taa. Asiakassovelluksen tehtäväksi jää silloin ensisijaisesti käyttöliittymän tarjoaminen käyttäjälle ja graafin piirtäminen. Kun asiakassovellus toimii kevyellä datamäärällä ja hakee tarvittaessa lisää dataa sovelluksen taustalla asynkronisesti, asiakassovelluksen käytöstä saa hyvinkin sulavan ja dynaamisen kokemuksen.

4. Esitettävän datan muodostus

Tässä luvussa tutkitaan datan esikäsittelyä tiedon siirtorajoitteiden ja tärkeyden näkökulmasta sekä esitetään oma ratkaisu kohdattuun ongelmaan.

4.1. Datan karsinta

Koska haluamme tarkastella signaalidataa nopeasti hitaan tietoverkon yli, ja esitettävän datan määrä voi olla huomattava, ei ole käytännöllistä käsitellä täyttä dataa. Esimerkiksi, jos laite tallentaa datapisteen kerran sekunnissa, ja pyydetty aika-alue on yksi kuukausi, datapisteitä on jo 2 628 000 kappaletta.

Kaiken signaalin analysoinnin perustana on se, että analysoitava data vastaa todellisuutta ja todellista lähdedataa. Matemaattiseen analyysiin tarvitaan usein absoluuttista täyttä dataa, jotta lopputulos on luotettava, tai on käytettävä karsintaan algoritmia, joka säilyttää kohdekäytölle oleellisen informaation.

Tässä tapauksessa analysoiva taho on ihmisen silmä, ja siten vaatimukset asettavat esitysteknologia ja silmän erotuskyky. Matemaattista esikarsintaa on tutkittu paljon ja tarkoitukseen on monia algoritmeja, mutta vain harva niistä tuottaa graafisesti piirrettynä visuaalisesti samannäköistä dataa.

Silmämääräisen prosessianalyysin mielenkiintoisin piste on usein yksittäinen poikkeava datapiste, joten näytteistys keskiarvolla tai mediaanilla ei ole hyväksyttävää, koska yksittäiset poikkeamat katoavat ja karsinnan tuote ei silmämääräisesti vastaa lähdedataa. Tilastollisen tiivisteen sijasta tärkeämpi on graafin selkeä visuaalinen muoto.

Koska tiedämme graafiesityksen olevan kaksiulotteinen kuva jostain mielivaltaisesta aika-alueesta, on esitystarkkuutta tehokkain rajoittaa aika-akselilla. Koska näkökyky on käyttäjäriippuvaista, oletetaan käyttäjän omaavan erittäin hyvän näön. Tämä korostaa esitysteknologian rajoitteiden merkitystä.

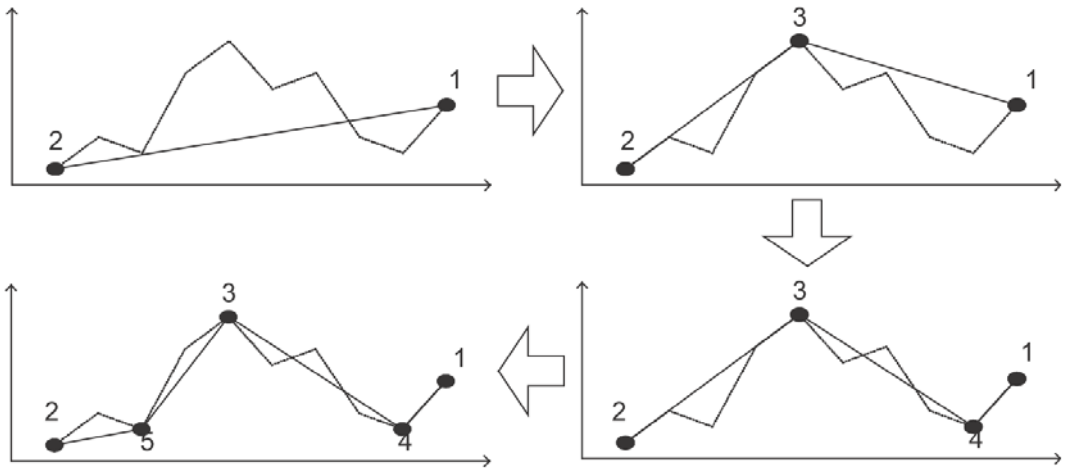
4.2. Karsintatapoja

Yksinkertaisin ratkaisu edellisessä kohdassa kuvattuun ongelmaan voisi olla piirtää signaaligraafi kuvaksi palvelimella ja siirtää se paikallisesti esitettäväksi. Vaikka tällä tavoin saataisiin visuaalinen kuva oikeasti täydestä datasta, staattinen kuva ei sisällä tietoa sisällöstään, kuten mikä signaali on kyseessä ja mitä sen arvot ovat milläkin kohdalla kuvassa, tai mikä signaali on valittuna, jos

käyttäjä haluaa tehdä sille operaatioita. Staattinen kuva myös pakottaisi käyttäjän odottamaan kaikkien toimintojensa suorittamista palvelimen päässä. Näistä syistä on järkevämpää piirtää graafi paikallisella sovelluksella, johon haetaan palvelimelta tarvittavat datapisteet, tarvittavalla karsitulla tarkkuudella.

Erilaisia datapisteiden karsintatapoja on tutkittu laajalti, mutta valtaosa esitetyistä ratkaisuista keskittyy datan esikäsittelyyn erityistä tilastollista tarkoitusta varten, tai muuten muodostaa datasta uusia tiivistettyjä arvoja, antaen tulokselle alkuperäisestä poikkeavia arvoja ja visuaalisen muodon. Tämä vaatimus sulkee pois kaikki parametriset karsinnat, kuten Fourier- tai wavelet-tekniikoihin perustuvat [Agrawal et al., 1993; Chan and Fu, 1999], sekä valta-osan datapisteisiin perustuvista algoritmeista [Fu et al., 2008b; Charbonnier, 2005; Keogh et al., 2001].

Eräs visuaalisesti saman muodon tuottava algoritmi on PIP (perceptually important point) [Fu et al., 2008a; Fu et al., 2008b]. Algoritmin toimintaperiaatetta on havainnollistettu kuvassa 1.



Kuva 1. PIP algoritmin alkutila ja kolme ensimmäistä kierrosta

Algoritmissa karsinnan lähtötilaksi otetaan lähdedatan ensimmäinen ja viimeinen datapiste. Sen jälkeen verrataan jokaisen datapisteen arvoa jo karsitun datan lineaarisesti interpoloituun vastineeseen ja lisätään suurimman etäisyyden datapiste karsittuun dataan. Tämä toistetaan kunnes suurin löydetty etäisyys on alle halutun tarkkuuden.

Tällaisella käsittelyllä saadaan datasta karsittua arvoina merkityksetön ja visuaalisesti näkymätön kevyt vaihtelevuus säilyttäen silti merkityksellinen data. Vaikka tämä algoritmi tuottaa halutun visuaalisesti vastaavan tuloksen, se ei ole sidottu tarkkuuteen ajassa, vaan lisää datapisteitä kunnes haluttu arvoja kuvaava tarkkuus on saavutettu. Tämä johtaa merkittävään datapisteiden ka-

saantumiseen alueille joissa signaali vaihtelee paljon. Pahimmassa tapauksessa karsittu data on sama kuin alkuperäinen täysi data.

Vaikka tämä algoritmi ei ole riittävä haluttuun tarkoitukseen, voi sitä silti hyödyntää datan esi- tai jälkikäsittelyyn ilman huomattavia negatiivisia vaikutuksia, kunhan toleranssit ovat tarpeeksi pieniä. Erityisen hyödyllinen se on mm. raskaiden tietokantojen pakkaamiseen siirtoa tai säilöä varten. Reaaliaikaiseen käyttöön algoritmi on kuitenkin rajoittavan hidas ($O(n^2)$) ja siten suositeltavaa vain kannan esikäsittelyyn.

Kaikki signaalilähteet eivät ole tyypillistä vaihtelevaa mittausdataa. Signaalilähde voi osoittaa tarkan arvon sijaan järjestelmän asetusta, jotain datasta esilaskettua portaistettua tilaa, tai signaalilähde saattaa olla jostain muusta syystä erityisen vakaa, kuten binääritilaa kuvaava signaali tai sammutustila.

Tällainen data sisältää paljon toistuvia muuttumattomia arvoja. Se on triviaali karsia poistamalla kaikki peräkkäisesti toistuvat arvot, säilyttäen vain ensimmäisen ja viimeisen datapisteen. Tämä toiminto on erittäin nopea, tuottaa täydellisen identtisen signaalimuodon ja on siten suositeltava ajaa aina ennen myöhempää käsittelyä.

4.3. Ehdotettu karsintatapa

Laajasta etsinnästä huolimatta tieteellisestä kirjallisuudesta ei löytynyt karsinta-algoritmia, joka säilyttäisi todelliset arvot, visuaalisen ulkomuodon sekä karsisi dataa tehokkaasti. Uskon kuitenkin löytäneeni ongelmaan yksinkertaisen, mutta tehokkaan ratkaisun.

Koska staattisen kaksiulotteisen digitaalisen kuvan pienin ulotteellinen yksikkö on yksi kuvapiste, on siitä käyttäjän tulkitsema käsitys myös samoin rajoitettu. Täten voidaan kuvan koosta (kuvapisteitä aika-akselin suuntaisesti) ja aika-akselin kattamasta ajasta laskea datan merkityksellinen näyteväli kaavalla

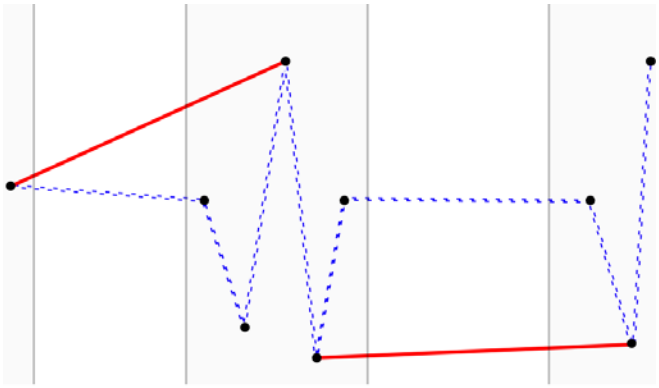
`näyteväli = ( loppuaika - alkuaika ) / kuvapisteiden määrä`.

Jos signaalin arvo vaihtelee yhden kuvapisteen näytevälin aikana, ei kuvasta ole tulkittavissa kuin arvon muutosväli. Täten voimme esittää näytevälien arvot viivana suurimman ja pienimmän arvon välillä Algoritmissa 1 esitetyllä tavalla. Tämän prosessin seurauksena on signaalista karsittu pisteet, jotka ovat viivaesityksessä käyttäjälle näkymättömissä, ja tulos sisältää vain kaksi datapistettä jokaista aika-akselin kuvapistettä kohti. Koska algoritmi ei muuta lähdedataa, näkyvät lähdedatan todelliset arvot edelleen sovellukselle karsinnan jälkeen. Algoritmin suoritus on myös erittäin nopea (O(n) ajassa).

```
data=täysi karsimaton data
näyteväli=( loppuaika - alkuaika ) / kuvapisteiden määrä.
karsittudata=tyhjä lista
valittuaika=alkuaika
toista kunnes ( valittuaika > loppuaika ) {
   näyte=data.aikaväli( valittuaika , valittuaika+näyteväli )
   karsittudata.lisää( suurin(näyte) )
   karsittudata.lisää( pienin(näyte) )
   valittuaika=valittuaika+näyteväli
}
karsittudata=poistaduplikaatit( karsittudata )
palauta karsittudata
```

Algoritmi 1. Min-max askeleellinen karsinta

On kuitenkin erityistilanteita, jolloin kuva ei vastaa todellisuutta. Erityistilanteet liittyvät datatiheyden vaihteluun. Kun datapisteiden välissä on yksi tai useampi näyteväli kokonaan ilman arvoja, saattaa datapisteiden välille virheellisesti piirtyä viiva poikki arvoalueen, jolla signaali ei todellisuudessa koskaan käy. Tätä on havainnollistettu kuvassa 2.



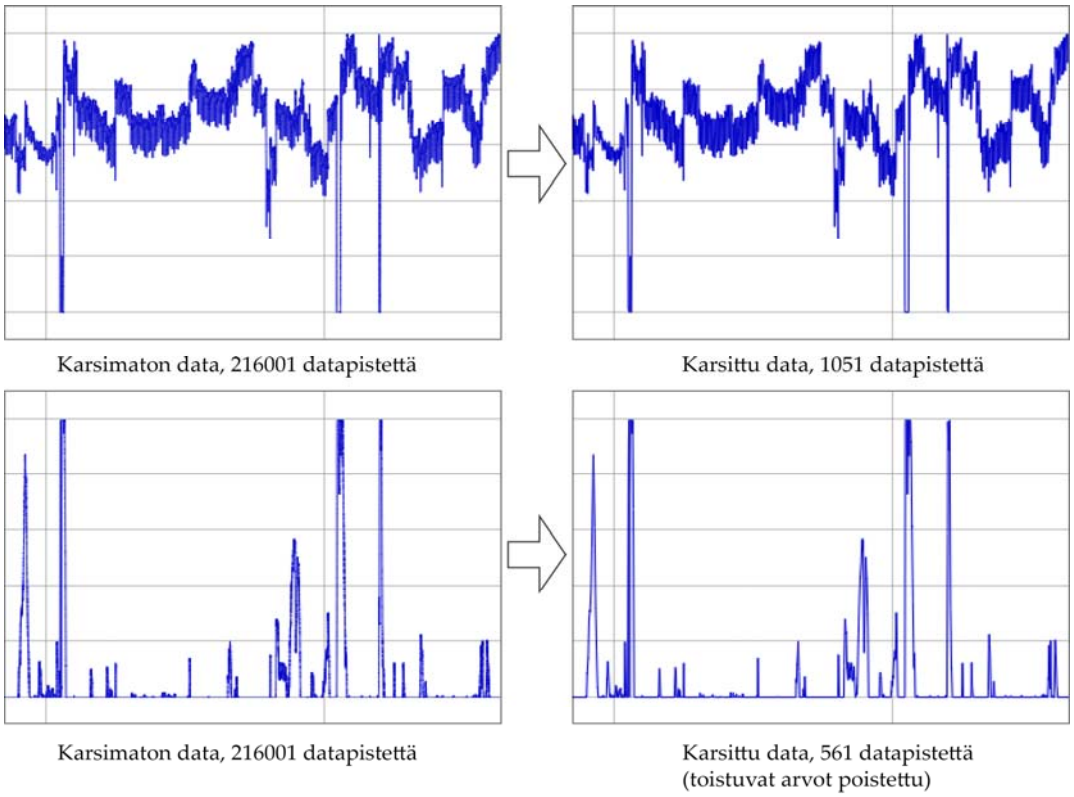
Kuva 2. Virheellinen karsinta tyhjien näytevälien yli

Kuvassa 2 on lähdedata kuvattu pisteinä ja pisteviivana. Datan näyteväli on kuvattu harmailla pystyviivoilla. Tyhjien näytteiden yli virheellisesti karsitut viivat on kuvattu punaisina suorina viivoina.

Tämä voidaan estää ottamalla tyhjää näytettä edeltävä ja seuraava piste mukaan, Algoritmissa 2 kuvattuun tapaan. Tällä tavoin tuotettu karsinta on kaikissa tapauksissa lähdedatan muotoista dataa.

```
data=täysi karsimaton data
näyteväli=( loppuaika - alkuaika ) / kuvapisteiden määrä.
karsitutindeksit=tyhjä lista
tyhjänäyte=epätosi
indeksi=0
näytealku=0
näyteloppu=0
näyteaika=data[0][‘aika’]
toista jos ( näytealku < pituus(data) ) {
  toista jos ( näyteaika+näyteväli < data[indeksi][‘aika’]
   ja indeksi < pituus(data) ) {
    indeksi=indeksi+1
  }
  näyteloppu=indeksi-1
  jos ( näytealku < näyteloppu ) {
    tyhjänäyte = epätosi
    näyte=data indeksistä näytealku indeksiin näyteloppu
    karsitutindeksit.lisää( suurin(näyte) )
    karsitutindeksit.lisää( pienin(näyte) )
    tyhjänäyte=epätosi
  } muuten jos ( data[näytealku][‘aika’] >= näyteaika ) {
    jos (tyhjänäyte=epätosi ja näytealku>0 ) {
      karsitutindeksit.lisää( näytealku -1 )
      tyhjänäyte=tosi
    }
    karsitutindeksit.lisää( näytealku )
  }
  näytealku=indeksi
  näyteaika=näyteaika+näyteväli
}
karsitutindeksit=poistaduplikaatit( karsitutindeksit )
karsittudata=data indekseistä karsitutindeksit
palauta karsittudata
```

Algoritmi 2. Vaihtuvan datatiheyden min-max askeleellinen karsinta



Kuva 3. Karsintatavan visuaalinen vertaus

Kuvassa 3 verrataan karsintatavan ulkonäköä ja tehokkuutta käsittelemättömään lähdedataan kahdella hyvin erilaisella signaalilla, joissa molemmissa on hyvin tiheää dataa. Vasemmanpuoleisessa kuvassa on signaalidata sellaisena kuin se signaalilähteestä saadaan, ja oikealla on signaalit karsinnan jälkeen. Alempi signaali on huomattavan paljon datasta 0-arvossa, ja siten toistuvien arvojen poiston hyödyllisyys näkyy erityisen hyvin karsintatehokkuudessa. Algoritmi on kuitenkin rajoittunut kuvapisteiden kokoon, ja monet modernit viivojen piirtotavat piirtävät viivat pehmennettynä, jäljitellen tarkempaa pistetarkkuutta. Kun kuvaa 3 katsoo tarkasti tai vertaa ohjelmallisesti, erottuu siinä tästä johtuvia poikkeamia. Nämä poikkeamat ovat kuitenkin silmämääräisesti lähes huomaamattomia ja siten hyväksyttäviä.

5. Haun tarkkuuden määritys ja datan tallennus

Tässä luvussa kuvataan datan karsintakyselyn muodostamista ja karsitun datan tallennusta käytännön näkökulmasta.

5.1. Halutun datatarkkuuden laskenta

Luvussa 4 esitetyn visuaalisesti ilmeen säilyttävän karsinnan toteuttamiseen tarvitaan jokaiselle aikaa kuvaavalle kuvapisteelle vähintään yksi aikanäyteväli, jolle algoritmia sovelletaan. Tämän voi yksinkertaisuudessaan laskea kaavalla

näyteväli = (loppuaika - alkuaika) / aika-akselin kuvapisteiden määrä.

Kertaesitykselle tämä on riittävä, mutta koska ladattua dataa täytyy pystyä tallentamaan sekä ottamaan huomioon tallennettu tarkkuus, ei ole käytännöllistä kirjata ylös jokaista tarkkuustasoa. Koska data tarvittua tarkempana on visuaalisesti samannäköistä, voidaan tarkkuus portaistaa kaavalla

$$\text{näyteväli} = 2^{\text{floor(log2(näyteväli))}}.$$

Tämän tuloksena tulee tarkempaan suuntaan pyöristetty tarkkuus, joka on vähintään yksi kuvapiste on yksi näyteväli, ja pahimmillaan yksi kuvapiste on kaksi näyteväliä. Tämä tarkoittaa sitä, että 1000 kuvapisteen aikajanaa esittävälle kuvalle ladattaisiin pahimmillaan 4000 datapistettä.

5.2. Datan tallennusrakenne

Kun signaalidataa on kerran ladattu, se on hyvä ottaa talteen, ettei seuraavalla kyselyllä tarvitse kysyä samaa tietoa uudestaan. Tätä dataa täytyy kuitenkin pystyä selaamaan ja katselemaan eri tarkkuuksilla. Tällöin täytyy kutsua olemassa olevaa tarkempaa dataa, joten täytyy myös merkitä, miten tarkkaa dataa on ladattu ja miltä alueilta se on saatu. Tässä kohdassa kuvataan tallennukseen kaksi hieman erilaista tapaa, yhdistetty ja eritelty tallennus.

5.2.1. Yhdistetty tallennus

Koska hetkellistä tarkkuustasoa tarkempi data on visuaalisesti identtistä, on yksinkertaisinta yhdistää ladattu data yhteen listaan. Samalla säästytään tietojen myöhemmin yhdistelyltä. Jos tarpeettoman tarkkaa dataa kertyy liikaa, sitä voi karsia uudestaan jollain luvussa 4 esitetyllä tavalla. Ladatusta datasta voi muodostaa tarkkuustasokartan soveltamalla yllä esiteltyä portaistuksen kaavaa. Tällöin ladatut alueet voi tallentaa tietorakenteeseen tarkkuudella Algoritmissa 3 kuvatulla menettelyllä.

```
tarkkuustaso = floor( log2( näyteväli ) )
uusidata=uusi ladattu data
uusilataus=uuden ladatun datan ensimmäinen ja viimeinen aikaleima
ladattudata=ladattudata.lisää(uusidata)
ladatutalueet[tarkkuustaso].lisää(uusilataus).
```

Algoritmi 3. Yhdistetty tarkkuuskartan tallennus.

Nyt voidaan uutta dataa kysyttäessä rakenteelle suorittaa Algoritmissa 4 esitetty kysely.

```
tarkkuustaso = floor( log2( näyteväli ) )
uusialue = haluttu aika-alue
toista jos ( tarkkuustaso > 0 ) {
   uusialue.poista_aikaväli( ladatutalueet[tarkkuustaso] )
   tarkkuustaso = tarkkuustaso – 1
}
palauta uusialue
```

Algoritmi 4. Tarkkuuskartan puuttuvan datan kysely.

Algoritmin lopputuloksena saadaan lista aika-alueista, joita ei ole ladattu nykyisellä tai tarkemmalla tarkkuustasolla ja jonka pohjalta voidaan muodostaa kantakysely.

5.2.2. Eritelty tallennus

Ladatun datan voi tallentaa myös erikseen tarkkuustasojen mukaan. Toimintatapa on hyvin samanlainen kuin yhdistetyssä tallennuksessa, mutta datan tallennus ja lukeminen täytyy käsitellä tarkkuustason mukaan. Tallennus onnistuu pienellä Algoritmi 3 muutoksella. Muutettu menettely on Algoritmi 5.

```
tarkkuustaso = floor( log2( näyteväli ) )
uusilatausalue=uuden ladatun datan ensimmäinen ja viimeinen aikaleima
uusidata=uusi ladattu data
ladatutalueet[tarkkuustaso].lisää(uusilatausalue)
ladattudata[tarkkuustaso].lisää(uusidata)
```

Algoritmi 5. Eritelty tarkkuuskartan tallennus.

Tällä tallennustavalla on se etu, että muistin loppuessa on helpompi karsia tarpeetonta dataa, ilman karsinta-algoritmeja. Yksittäisen tarkkuustason poistaminen onnistuu helposti seuraavasti:

```
ladattudata[tarkkuustaso]=tyhjä
ladatutalueet[tarkkuustaso]=tyhjä.
```

Ennen tarkkuustason poistoa on kuitenkin hyvä tarkistaa Algoritmilla 4, tuleeko dataan aukkoja. Aukot voi korjata karsimalla tarkemmasta datasta jollain

luvussa 4 mainitulla menetelmällä tai säilyttää aukkojen kohdalle sattuvat datapisteet aluetietoineen. Tarkkaa dataa on käytännössä hyvin harvoin alueella, jossa sitä ei jo olisi tallennettuna harvemmalla tarkkuudella, koska signaalia yleensä selataan laajemmassa yleisnäkymässä, ja sitten tarkennetaan mielenkiintoisiin kohtiin.

6. Yhteenveto

Tässä tutkielmassa tutkittiin teollisuuden signaalidatan visuaaliseen analysointiin tarvittavia teknologioita käytännön näkökulmasta, ja todettiin selkeä asiakas-palvelin -arkkitehtuuri välttämättömäksi.

Palvelimella tapahtuvaan datan karsintaan tutkittiin lukuisia algoritmeja, joista yksikään ei vastannut vaatimuksia tarkan visuaalisen muodon säilymiseen ja datan tehokkaaseen karsintaan. Karsintaongelmaan ehdotettiin oma ratkaisu, Algoritmi 2. Ehdotettu karsinta-algoritmi tuottaa lähes kaikissa tapauksissa alkuperäistä täyttä dataa vastaavan visuaalisen muodon ja todelliset datapisteiden arvot. Koska palvelimen karsinta-algoritmi on keskeisessä osassa, olisi se hyvä saada mahdollisimman lähelle lähdedataa, jopa tietokannan sisälle SQL-lauseena jos mahdollista.

Yksi karsinta-algoritmin puute on toistuva samojen datapisteiden siirto: algoritmi ei tiedä, mitä dataa asiakkaalla on, vaan jokainen kutsu lähettää kaikki alueen pisteet pyydetyllä tarkkuudella. Jatkotutkimuksessa tähän voisi puuttua. Kerralla siirrettävä pisteiden määrä on kuitenkin nykyisessä tilanteessakin suhteellisen alhainen, ja lisäprosessointi palvelimella täytyy tasapainottaa ylimääräisistä pisteistä syntyvään siirtoviiveeseen.

Lisäksi esitettiin kaksi tapaa tallentaa ladattu data asiakassovelluksessa. Mainitut tallennustavat ovat hyvin yksinkertaisia ja niitä voisi jatkotutkimuksessa parantaa mm. ottamalla eri tarkkuustasot paremmin huomioon ja välttää toistuvaa dataa.

Tutkielmassa kuvattuja huomioita ja algoritmeja on toteutettu käytännön sovellukseksi, joka lukee reaaliaikaista signaalidataa tietokannasta, karsii sen ja esittää asiakassovelluksessa käyttäjälle. Lähestymistapa on tässä sovelluksessa todettu aiempia sovelluksia huomattavasti sulavammaksi ja käyttäjälle miellekkäämmäksi. Itse karsinta tapahtuu käytännössä reaaliajassa ja pullon-kaulaksi muodostuu tietokantakysely ja mahdolliset verkon viiveet.

Teollisuudessa tehdään paljon analyysiä tässä tutkielmassa esitellyn 1-ulotteisen graafiesityksen lisäksi myös 2-ulotteisesta profiiliesityksestä, jossa ilmenee hyvin paljon samoja ongelmia. Näihin ongelmiin voidaan hyödyntää paljon tässä tutkielmassa tehtyjä huomioita ja algoritmeja, mutta asia vaatii myös lisä-

tutkimusta. Käsiteltävän datan määrä on moninkertainen ja siten datakarsinnan tehokkuus myös tärkeämpi.

Viiteluettelo

[Agrawal et al., 1993] Rakesh Agrawal, Christos Faloutsos and Arun Swami (1993). Efficient Similarity Search in Sequence Databases. In: *Proceedings of the 4th Conference on Foundations of Data Organization and Algorithms*, 69–84.

[Chan and Fu, 1999] Kin Pong Chan and Wai-Chee Fu (1999). Efficient Time Series Matching by Wavelets. In: *Proceedings of the 15th IEEE International Conference on Data Engineering*, 126–133.

[Charbonnier, 2005] Charbonnier S., On line extraction of temporal episodes from ICU high-frequency data: A visual support for signal interpretation. *Computer Methods and Programs in Biomedicine* **78** (2005), 115—132.

[Eaton et al., 2012] John W. Eaton, et al. (2012). GNU Octave (Version 3.6.2) [Software] Available from http://www.gnu.org/software/octave/.

[Fu et al., 2008a] Tak-chung Fu, Fu-lai Chung, Ka-yan Kwok and Chak-man Ng, Stock time series visualization based on data point importance. *Engineering Applications of Artificial Intelligence* **21** (2008) 1217–1232.

[Fu et al., 2008b] Tak-chung Fu, Fu-lai Chung, Robert Luk and Chak-man Ng, Representing financial time series based on data point importance. *Engineering Applications of Artificial Intelligence* **21** (2008) 277–300.

[Keogh et al., 2001] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani and Sharad Mehrotra, Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems* **3** (2001) 263–286.

[Mathworks, 2012] Mathworks (2012). Mathworks MATLAB (Version R2012b).

Käyttäjäkokemus verkossa järjestetyllä etäkurssilla

Ville Hämäläinen

Tiivistelmä.

Tässä tutkielmassa käsitellään informaatiotieteiden yksikön järjestämän Principles of Usability, User Experience and User Interfaces -kurssin käyttäjäkokemusta. Käyttäjäkokemuksella viitataan käyttäjien käsityksiin ja kokemuksiin, joita on syntynyt tuotteen, järjestelmän tai palvelun käytön yhteydessä. Tässä tapauksessa käyttäjäkokemus piti sisällään PUXUI-kurssille osallistuneiden kokemuksia kurssin ja oman oppimisensa onnistumisesta. Positiivinen käyttäjäkokemus vaikuttaa käyttäjien haluun jatkaa oppimisympäristön käyttöä ja sen on todettu vaikuttavan myös oppimisen onnistumiseen. Kurssi oli toteutettu Moodlessa, joka on sähköinen oppimisympäristö eli informaatiojärjestelmä, joka auttaa e-oppimista. Sähköiset oppimisympäristöt tukevat usein yhteistyötä käyttäjien välillä, mutta PUXUI-kurssilla tätä mahdollisuutta ei hyödynnetty.

PUXUI-kurssin käyttäjäkokemus muodostui alku- ja välikyselyn perusteella positiiviseksi. Huolimatta siitä, että kurssi järjestettiin englanniksi, joka ei ollut enemmistön äidinkieli, myös suurin osa itsensä vasta-alkajiksi HCI-alalla määritelleistä läpäisivät tentin. Vain kolme alkukyselyyn vastannutta jätti osallistumasta tenttiin. Hyvinä puolina pidettiin kiinnostavaa aihetta, monipuolisia materiaaleja ja sähköisen etäkurssin tarjoamaa joustavuutta. Huonoina puolina mainittiin pieni joukko teknisiä ongelmia, välittömän palautteen puute ja harjoitusten työläys. Tutkielman lopussa käsitellään sähköisten oppimisympäristöjen etuja ja haasteita sekä tehdään muutama ehdotus PUXUI-kurssin käyttäjäkokemuksen parantamiseksi tulevaisuudessa.

Avainsanat ja -sanonnat: Käyttäjäkokemus, sähköiset oppimisympäristöt, käytettävyys, vuorovaikutus.

CR-luokat: K.3.1

1. Johdanto

Internet ja tietotekniikka mahdollistavat opetuksen tarjoamisen ja oppimisen ajasta, paikasta tai molemmista riippumatta. Perinteisillä staattisilla verkkosivuilla voidaan tarjota oppaita ja ohjeita tekstin, kuvien, äänen sekä videoiden avulla. Laajakaistayhteydet mahdollistavat luentojen järjestämisen videokonfe-

rensseina tai live-videona verkon yli. Tällaista opetusta ei rajoita luentosalien koko tai luennoitsijan maantieteellinen etäisyys luennolle osallistujiin, sillä ne eivät ole oleellisia asioita tietoverkoissa. Vähintäänkin teorian tasolla tieto on kaikkien niiden saatavilla, joilla on pääsy internetiin yhteydessä olevalle laitteelle.

Suoritin lukioni pienellä paikkakunnalla ja osallistujien pienen määrän vuoksi elämänkatsomustiedon pakolliset kurssit järjestettiin verkossa yhteistyönä muutaman muun pienen lukion kanssa. Kurssi toteutettiin siten, että luimme oppikirjasta tietyt luvut, jonka jälkeen osallistuimme pohdintatehtävien avustuksella keskusteluun kurssin keskustelufoorumilla. Verkossa järjestetty kurssi teki maantieteellisestä etäisyydestä merkityksettömän seikan ja suurempi ryhmäkoko mahdollisti mielekkäämmän keskustelun ja mielipiteiden vaihdon kuin kahden tai kolmen hengen ryhmässä olisi ollut mahdollista. Tuohon aikaan laajakaistayhteydet olivat vasta yleistymässä, joten verkossa tarjottu materiaali rajoittui lähinnä tekstiin ja pieniin kuviin.

Nykyiset sähköiset oppimisympäristöt tarjoavat laajemman valikoiman työkaluja verkko-opetukseen. Tampereen yliopistolla on käytössä Moodle-oppimisympäristö, jota käytetään ainakin osalla Tampereen yliopiston tarjoamista kursseista. Osalla kursseista Moodlea käytetään materiaalien jakamiseen ja tehtävien palauttamiseen, mutta on myös kursseja, jotka järjestetään kokonaisuudessaan Moodlessa. Informaatiotieteiden yksikön järjestämä Principles of Usability, User Experience and User Interfaces -kurssi (PUXUI) on kokonaan Moodlessa etänä suoritettava opintojakso. Luennot on toteutettu diaesityksinä, joihin on liitetty luennoitsijan ääniraita. Kurssin läpipääsy edellyttää Moodlessa tehtävien harjoitusten suorittamista ja tentin läpäisyä. Perinteistä kasvokkain tapahtuvaa opetusta ei kurssilla järjestetä.

PUXUI järjestetään ensimmäistä kertaa tänä vuonna ja toimin kurssilla assistenttina lukuvuonna 2012-2013. Minulla oli erinomainen tilaisuus tarkkailla kurssin etenemistä ja selvittää osallistujien mielipiteitä ja kokemuksia kokonaan verkossa järjestettävästä kurssista. Olen kiinnostunut selvittämään, millaiseksi muodostui tämän Moodle-kurssin käyttäjäkokemus.

Aloitan määrittelemällä, mitä käyttäjäkokemuksella tarkoitetaan ja mitä ovat sähköiset oppimisympäristöt. Sen jälkeen tarkastelen aikaisempien tutkimusten valossa ja osallistujien palautteen perusteella PUXUI-kurssin käyttäjäkokemusta. Lopuksi käyn läpi tutkitun kaltaisen verkkokurssin vahvuuksia ja heikkouksia ja mahdollisia kehityskohteita.

2. Sähköiset oppimisympäristöt ja käyttäjäkokemus

Sähköisillä oppimisympäristöillä tässä tutkielmassa viitataan yhteistyötä tukevaan virtuaaliseen ympäristöön (collaborative virtual environment), jossa opetus tapahtuu. Tällainen ympäristö on tietotekniikkaan perustuva, virtuaalinen tila tai joukko paikkoja. Tässä tilassa tai näissä paikoissa ihmiset voivat tavata ja olla vuorovaikutuksessa toistensa, agenttien tai muiden virtuaalisten objektien kanssa. Virtuaalisilla ympäristöillä voidaan tarkoittaa täysin 3D-mallinnetuista ympäristöistä puhtaasti tekstipohjaisiin. [Konstantinidis et al., 2009]

Pedagogisia hyötyjä yhteistyötä tukevilla virtuaalisilla oppimisympäristöillä on paljon. Opiskelijat voivat selventää hyvin abstraktia tai vaikeasti määriteltyä tietoa sekä kiinnittää huomiota avoimiin ongelmiin. Yhteistyö myös mahdollistaa myös keskustelun vaikeiden ongelmien eri näkökulmista ja ongelmanratkonnan yhteistyössä muiden osallistujien kanssa. Suurin etu on kuitenkin vuorovaikutus, joka parhaimmassa tapauksessa kannustaa aktiivisuuteen, tekee oppimisesta realistisempaa ja lisää motivaatiota. Virtuaalisessa ympäristössä tapahtuvan vuorovaikutuksen on myös huomattu tasapainottavan opettajan ja oppilaan välistä suhdetta ja kaventavan sukupuolten välistä eroa. [Konstantinidis et al., 2009]

Toinen tapa määritellä virtuaalinen oppimisympäristö on todeta, että se on informaatiojärjestelmä, joka auttaa e-oppimista [Lin, 2012]. Tämä vastaa enemmän tässä tutkielmassa käsiteltyä PUXUI-kurssia, sillä vuorovaikutus muiden kurssilaisten välillä oli tällä kurssilla olematonta. Tämä johtuu pääosin tavasta, jolla kurssi ja sen harjoitukset oli toteutettu.

Verkossa tapahtuvan etäopetuksen etuna pidetään matalia kustannuksia ja perinteistä kurssia laajempaa saavutettavuutta. Kuitenkin monilla etäkursseilla keskeyttäneiden määrät ovat korkeita ja usea kurssi jättääkin opiskelijat turhautuneiksi tai innottomiksi [Notess, 2001]. Mikäli käyttäjäkokemus on positiivinen, opiskelijat ovat motivoituneempia käyttämään sähköistä oppimisympäristöä ja osallistumaan siellä tapahtuvaan opetukseen.

Petre ja muut [1997] vertailivat kirjekurssina suoritettua etäkurssia, jossa jokainen opiskelija sai avukseen henkilökohtaisen ohjaajan, verkossa toteutettuun kurssiin, jossa vuorovaikutus tapahtui kaikille opiskelijoille avoimella foorumilla sekä sähköpostin välityksellä. Perinteisesti toteutetulla kurssilla ohjaajat saattoivat myös järjestää halukkaille ohjattavilleen ryhmätapaamisia, mutta verkossa järjestetyllä kurssilla tämä vuorovaikutusmuoto puuttui. Saatujen tulosten perusteella he totesivat, että näiden kahden eri tavoin toteutetun etäkurssin tuloksia oli kuitenkin mahdollista vertailla keskenään.

Ohjaajat kertoivat, että sähköpostin välityksellä viestintä oli avoimempaa kuin se oli perinteisesti järjestetyllä kurssilla puhelimen välityksellä. Vuorovaikutus oli vilkkaampaa verkkokurssilaisten kanssa ja varsinkin epävarmemmat opiskelijat kokivat sähköpostin vähemmän rajoittavaksi viestintätavaksi. Kuitenkin verkkokurssille osallistuneet opiskelijat kertoivat useammin olleensa kanssaopiskelijoidensa panokseen pettyneempiä kuin perinteisellä tavalla järjestetylle etäkurssille osallistuneet. On myös huomioitava, ettei moni verkkokurssille osallistunut olisi päässyt osallistumaan kasvokkain tapahtuviin ryhmätapaamisiin. Tutkijat huomasivat, että tyytyväisyys kurssiin riippui opiskelijoiden ennakko-odotuksista kurssia kohtaan. Onnistuneen oppimisen kannalta opetustapaa oleellisemmaksi seikaksi osoittautui opiskelijoiden käsitys omasta kyvystään ja osaamisestaan.

ISO 9241-110:2010 -standardi määrittelee käyttäjäkokemuksen olevan ”a person's perceptions and responses that result from the use and/or anticipated use of a product, system or service”. Käyttäjäkokemus vaikuttaa tämän määritelmän perusteella yhteen käytettävyyden mittariin, tyytyväisyyteen. Sähköisen oppimisympäristön käytettävyys on riippuvaista myös siitä, millaiseksi käyttäjät kokevat sen käytön ja miten he kokevat oppimisympäristön käytön vastaavan odotuksiaan [Law and Sun, 2012]. On hyvä myös huomata, että tällä tavoin määritelty käyttäjäkokemus vaikuttaa oleellisesti opiskelijoiden oppimiseen. Positiivinen käyttäjäkokemus vahvistaa opiskelijoiden positiivisia tulkintoja omista kyvyistään ja osaamisestaan.

Sähköiset oppimisympäristöt voidaan nähdä tuotteina, joiden käyttötarkoitus on tarjota työkalut opetuksen ja oppimisen mahdollistamiseksi. Tuotteita ei kuitenkaan käytetä pelkän instrumentaalisen arvonsa vuoksi. Mikäli käyttäjät kokevat tuotteen itselleen hyödylliseksi, he ovat halukkaampia käyttämään sitä useammin kuin kerran [Karapanos et al., 2009]. Käyttäjien kokemus tuotteesta on siis yhtä tärkeä kuin tuotteen instrumentaalinen arvo ja helppokäyttöisyys.

Halu käyttää tuotetta uudelleen on sähköisten oppimisympäristöjen kannalta tärkeää. Koska oppimisympäristöjä käytetään yleensä jo yhden kurssin aikana useaan kertaan, oppimisympäristön menestys on enemmän riippuvainen pitkän ajan käytöstä kuin ensimmäisestä käyttökerrasta. Tutkimukset osoittavat, että käyttäjien kokemus sähköisen oppimisympäristön sopimisesta heidän omiin tarpeisiinsa ja tyytyväisyys saattavat suoraan vaikuttaa oppimiseen ainakin tehokkuuden ja tuottavuuden suhteen. [Lin, 2012]

Käyttäjäkokemus ja sähköiset oppimisympäristöt liittyvät toisiinsa oppimisen kautta. Verkon kautta tapahtuva opetus poikkeaa kokemuksena perinteisestä opetuksesta. Opiskelijat täytyy pystyä pitämään kiinnostuneina opetuk-

seen liittyvästä materiaalista, opiskelijoiden pitää pystyä navigoimaan tehokkaasti oppimisympäristön materiaalien, työkalujen sekä aktiviteettien joukossa ja heidät pitää saada osallistumaan aktiivisesti oppimiseen liittyviin toimiin. [Notess, 2001]

Richards ja Kelaiah [2012] tutkivat, mitkä ovat oppimisen ja käyttäjäkokemuksen kannalta tärkeitä ominaisuuksia virtuaalisissa oppimisympäristöissä. He huomasivat, että oppimisympäristön käytön opetteluun kuluva aika, suoritusten tehokkuus, käyttötapojen mieleen palauttaminen pitkän ajan kuluttua ja käyttäjän tekemien virheitten määrä ovat yhtä tärkeitä käytettävyyden mittareita niin virtuaalisissa oppimisympäristöissä kuin muissakin tietoteknisissä järjestelmissä. Jotta opiskelijat voisivat keskittyä opittavan asian opetteluun, tulisi oppimisympäristön käyttö olla mahdollisimman sujuvaa, yksinkertaista ja kivutonta. Jos virtuaalisen oppimisympäristön käyttö vaatii pitkän opettelujakson ja vielä senkin jälkeen se on monimutkaista ja hidasta, opiskelijat joutuvat käyttämään oppimisen lisäksi energiaansa oppimisympäristön käyttöön.

3. Käyttäjien kokemuksia PUXUI-verkkokurssista

Kurssilaisilta kyseltiin kurssin alussa heidän ennakkotietojaan kurssin aiheista ja etäkursseista, motivaatiotaan ja toiveitaan kurssiin liittyen. Puolivälissä kurssia kyseltiin kokemuksia kurssin sisällöstä siihen mennessä ja toiveita loppukurssiin liittyen. Kyselyt toteutettiin sähköisinä lomakkeina, jotka sisälsivät monivalintakysymyksiä sekä avoimia kysymyksiä. Alkukyselyyn vastasi 31 kurssilaista ja puolivälin kyselyyn vastasi 21 kurssilaista.

Kurssilla käytetty kieli materiaaleissa, tehtävissä ja kokeessa oli englanti. Suurimmalle osalle PUXUI-kurssin osallistujista englanti ei kuitenkaan ollut äidinkieli. Tämä on varmasti luonut oman haasteensa niille, joilla englanti ei ole vahvin kieli. Osa harjoituksissa käytetystä materiaalista oli tieteellisiä artikkeleita, joiden lukeminen on vaativaa hyvälläkin englannin kielen taidolla.

Kurssin harjoitukset olivat pääasiassa toteutettu kyselyinä, joihin opiskelijat annettujen materiaalien ja luentodiojen pohjalta vastasivat. Harjoitusten määräajan jälkeen vastauksista tehtiin yhteenvedot, jotka julkaistiin keskustelufoorumilla, mutta yksikään yhteenveto ei avannut keskustelua, vaikka mahdollisuus siihen olisi ollut.

Alkukyselystä selvisi, että hieman alle puolella kurssilaisista (45%) ei ollut aikaisempaa kokemusta oppimisympäristö Moodlen käytöstä ja noin puolella (52%) ei ollut minkäänlaista aikaisempaa kokemusta etäkursseista yleisesti. PUXUI-kurssin osallistujat jakautuivat siis puoliksi niihin, joilla ei ollut minkäänlaisia ennakkotietoja verkossa suoritettavasta etäopiskelusta, ja niihin,

joilla oli jonkinlainen mielikuva siitä, mitä kyseisen kaltainen opiskelu pitää sisällään. Tältä pohjalta on tietenkin vaikea sanoa, kuinka positiivisia odotuksia opiskelijoilla oli kurssin suhteen. Kuitenkin aiempaa kokemusta omaavilla nuo odotukset varmasti olivat realistisempia kuin niillä, joilta aiemmat kokemukset puuttuivat kokonaan.

Vain vajaalla neljänneksellä (23%) ei ollut aiempaa kokemusta kurssin aihealueesta, reilulla puolella kurssilaisista (55%) oli jonkin verran kokemusta ja lopuilla oli paljon kokemusta opintojensa puolesta tai he olivat työskennelleet alalla. Vaikka siis puolelle kurssilaisista etäkurssi oli uusi opiskelumuoto, vain vajaalle neljännekselle opetettava asia oli täysin uutta. Voidaan siis perustellusti olettaa, että suurimmalla osalla kurssilaisista oli jo pohjatietoa, johon uutta asiaa pystyi suhteuttamaan ja soveltamaan. Verkossa järjestetty etäkurssi vaatii kuitenkin opiskelijoilta suurempaa vastuuta omasta oppimisestaan kuin perinteisemmät opetusmenetelmät [Petre et al., 1997]. Niinpä tarvittavien tietojen hankkiminen kysymällä kurssin vetäjiltä, lukemalla alan kirjallisuutta tai jollain muulla tavalla, oli opiskelijoista itsestään kiinni. Taulukossa 1 näkyy, moniko itsensä aloittelijaksi tai kokeneeksi määritellyt läpäisi tentin ja moniko sai hylätyn arvosanan. Kolme alkukyselyyn vastanneista jätti osallistumatta syksyn tenttiin. Tenttiin osallistumatta jättäneet olivat kuitenkin opiskelijoita, joiden ei ollut syystä tai toisesta tarpeen osallistua tenttiin.

	Vasta-alkaja HCI-alalla	**Kokemusta HCI-alalla**
Läpäisi tentin	18	5
Ei läpäissyt tenttiä	3	2

Taulukko 1. Syksyn PUXUI-kurssin tenttiin osallistuneiden menestys

Noin puolille kurssilaisista (52%) PUXUI ei ollut tutkintoon vaadittu kurssi. Kysyttäessä odotuksia ja kiinnostusta kurssia kohtaan vain muutama mainitsi osallistuvansa kurssille pelkkien opintopisteiden takia. Tästä voidaan olettaa, että opiskelijat olivat ainakin kurssin alussa motivoituneita oppimaan kurssin asioita.

Puolivälikyselyssä pyydettiin kurssilaisia aluksi kertomaan kaksi hyvää ja kaksi huonoa asiaa. Kolme teemaa nousi esille hyviksi koettuina asioina: kurssin aihe, oma etenemistahti ja materiaalit. Moni mainitsi kurssin aiheen olevan mielenkiintoinen. Tämä toistaa alkukyselyssäkin ilmennyttä kiinnostusta kurssin aihetta kohtaan. On tietysti huomattava, että välikyselyyn jätti vastaa-

matta noin kolmannes kurssilaisista, jotka vastasivat alkukyselyyn, joten vastauksiin on saattanut painottua motivoituneimpien opiskelijoiden mielipiteet.

Toisena hyvänä asiana koettiin kurssilaisten mahdollisuus vaikuttaa omaan aikatauluunsa ja työtahtiinsa. Vaikka harjoituksien tekemiseen oli takarajat, pystyivät opiskelijat itse valitsemaan, milloin tehtävät tekivät. Suurimman osan ajasta kurssilaisten oli mahdollista tehdä seuraavankin viikon tehtäviä ennakkoon. Tämä mahdollisti suuren joustavuuden aikataulutuksen suhteen. Opiskelijat pystyivät myös käyttämään luentomateriaalien opetteluun itse valitsemansa ajan. Heidän oli mahdollista lukea ja kuunnella uudelleen ne kohdat, joita eivät olleet ymmärtäneet tai siirtyä eteenpäin, jos asiat vaikuttivat selviltä. Tämä mahdollisti nopeille oppijoille mahdollisuuden edetä nopeasti ilman, että hitaammat olisivat sen vuoksi joutuneet kiirehtimään.

Kolmanneksi onnistumisen aiheeksi nousi materiaalien monipuolisuus. Kurssin varsinainen luentomateriaali oli toteutettu nauhoitettuna diaesityksenä, johon oli liitetty luennoitsijan ääni. Tämän lisäksi kurssilla käytettiin harjoituksissa aiheisiin liittyviä tieteellisiä tekstejä, Microsoftin ja Applen tyylioppaita ohjelmistokehittäjille, videoita sekä aiheisiin liittyviä verkkosivustoja. Iso osa materiaaleista sitoi luentodioilla esiteltyjä asioita käytännön sovelluksiin ja prototyyppeihin. Kurssilaiset pitivät siitä, että luentokalvoilla esiteltyjä asioita käytiin läpi laajasti erityyppisillä materiaaleilla harjoitusten yhteydessä.

Negatiivisista kokemuksista nousi esille yhtenä suurena joukkona tekniset ongelmat. Muita toistuvia kommentteja oli avointen kysymysten vastausten pituusrajaus, välittömän palautteen puute ja harjoituksiin menevä aika. Kurssilaiset kokivat luentodiojen äänenlaadun olevan heikko ja tästä huomautettiin muutamaan kertaan. Diaesityksen mainittiin myös menevän epätahtiin nauhoitetun äänen kanssa, mikäli diaesityksessä palasi taaksepäin. Myös osa käytetyistä videoista toimi huonommin hitaammilla verkkoyhteyksillä. Osa koki vaikeaksi päästä käsiksi tieteellisiin teksteihin yliopiston ulkopuolisilta koneilta, jolloin piti käyttää Suomen korkeakouluissa käytössä olevaa Nelli-tiedonhakuportaalia.

Toinen ongelma oli puoliksi tekninen ja puoliksi kurssin vetäjien aiheuttama ongelma. Suuren vastausmäärän käsittelyn helpottamiseksi harjoituksissa olleiden avointen kysymysten vastausten pituus rajattiin tiettyyn merkkimäärään. Kysymykset osoittautuivat kuitenkin liian laajoiksi, jotta niihin olisi ollut mahdollista vastata annetulla merkkimäärällä. Kurssin opiskelijat kokivat tämän ongelmaksi ja loppuihin harjoituksiin vastausten mahdollista pituutta kasvatettiinkin huomattavasti.

Kolmas ongelma teknisten lisäksi oli välittömän palautteen puute. Kurssilaisten vastauksista kerättiin aina määräajan jälkeen yhteenvedot, joissa kerrattiin kysytyt kysymykset, kerrottiin mitä kurssilaiset olivat vastanneet ja mitä olivat ne oppimistavoitteet, joihin kysymyksillä pyrittiin. Moni kuitenkin koki, etteivät pystyneet seuraamaan omaa menestystään kurssin aikana ja olisivat kaivanneet mahdollisuutta tarkastella omia vastauksiaan myös harjoitukseen vastaamisen jälkeen. Tätä mahdollisuutta ei opiskelijoille ollut tarjolla johtuen tavasta, jolla harjoitukset oli teknisesti Moodlessa toteutettu. Opiskelijoiden oli myös vaikea päätellä harjoituksia tehdessään, kuinka suuri osa harjoituksesta oli vielä tekemättä.

Neljänneksi ongelmaksi muodostui aika, joka kurssilaisilla meni harjoitusten tekemiseen. Moni koki harjoitukset pitkiksi ja aikaa vieviksi. Eräs arvio oli kuuden tunnin työ viiteen harjoitukseen. On kuitenkin syytä muistaa, että yksi opintopiste vastaa 27 tunnin työpanosta ja PUXUI-kurssin suorittamisesta sai viisi opintopistettä. Harjoituksia koko kurssilla oli yhteensä vain 30, joten ansaittu opintopistemäärä vastaa varmasti harjoituksiin, luentomateriaaleihin ja tenttiin käytettyä aikaa. Todennäköisesti tästä asiasta huomauttaneet kurssilaiset eivät olleet ymmärtäneet, paljonko työtunteja viiden opintopisteen kurssi pitää sisällään. Kurssin aikana oli myös nähtävissä, että osa opiskelijoista jätti harjoitusten tekemisen viimeiseen iltaan, jolloin työmäärä saattaa tuntua suurelta.

Kun kurssilaisilta selvitettiin heidän mielipiteitään vuorovaikutuksesta kurssilla, kysyttiin ensin, kokivatko he tulleensa jätetyksi yksin kurssilla. Yli puolet vastasi, ettei ollut ajatellut koko asiaa ja hieman alle kolmasosa ei tuntenut tulleensa yksin jätetyksi. Kaikista vastanneista kurssilaisista 81% allekirjoitti väitteen, että PUXUI-kurssi on pääasiassa itsenäistä työtä, mutta se ei haittaa. Kolmasosa jäi kaipaamaan lisää vuorovaikutusta muiden kurssilaisten kanssa, mutta lähes saman verran kurssilaisia totesi saavansa palautetta vertaisiltaan muita reittejä.

Alkukyselystä selvisi, että kurssille osallistujat olivat ainakin omien sanojensa mukaan motivoituneita ja kiinnostuneita oppimaan kurssin asioita. Kurssille osallistui lähes yhtä monta sellaista oppilasta, joilla ei ollut kokemusta etäkursseista tai Moodlesta, kuin heitä, joilla kokemusta oli ainakin jonkin verran. Havainnoitavina oli siis melko heterogeeninen joukko. Aloittelijoiden ja kokeneempien käyttäjien kokemus poikkeaa odotusten suhteen toisistaan. Aloittelijoilla ei ole aikaisempia kokemuksia, joiden perusteella he osaisivat odottaa kurssin etenevän tietyllä tavalla. Tässä tutkielmassa ei eroa aloitte-

lijoiden ja kokeneempien välille ole kuitenkaan tehty, joten käyttäjäkokemus tässä tapauksessa on yleistys erilaisten käyttäjien yksittäisistä kokemuksista.

Positiivisia kokemuksia loivat kiinnostava aihe, kurssilaisten kyky vaikuttaa omaan etenemistahtiinsa ja harjoitusten tekohetkeen sekä monipuoliset materiaalit. Negatiivista palautetta sai joukko teknisiä ongelmia, vähäinen välitön palaute ja tunne harjoitusten työläydestä. Osa teknisistä ongelmista johtui kurssilaisten omista laitteista ja olisi ollut vältettävissä esimerkiksi käyttämällä yliopiston tietokoneluokkien koneita. Kokemukseen harjoitusten työläydestä vaikutti osittain myös se, ettei kurssilaisilla ollut mahdollisuutta tehtäviä tehdessään nähdä, montako tehtävää oli vielä jäljellä. Suurin syy vaikutti kuitenkin olevan se, että kurssilaiset yrittivät tehdä viikon edestä harjoituksia yhdessä illassa.

Välikyselyn perusteella kurssilaiset vaikuttivat enimmäkseen tyytyväisiltä siihen, miten kurssi oli siihen mennessä edennyt ja miten se oli toteutettu. Vaikka suora vuorovaikutus muiden kurssilaisten kanssa oli olematonta, vain kolmasosa välikyselyyn vastanneista olisi kaivannut sitä lisää. Ainoa kontakti kurssilaisten välillä olivat harjoitusten yhteenvedot, joissa osallistujat pääsivät lukemaan ja näkemään, miten harjoituksissa oli yleisesti pärjätty.

4. Yhteenveto

On selvää, että sähköisen etäkurssin onnistuminen riippuu suurelta osin käytetyn oppimisympäristön käytettävyydestä sekä siitä, miten hyvin kurssin materiaali ja opetus on suunniteltu sekä toteutettu. Hyvän kurssimateriaalin avulla opiskelija löytää etsimänsä tiedon helposti ja nopeasti. Tämä on tärkeää, sillä perinteisestä kasvokkain tapahtuvasta opetuksesta poiketen opiskelijalla ei ole mahdollisuutta saada nopeasti selitystä opettajalta kohtaan, jota ei aivan ymmärtänyt. Vaikeaselkoinen materiaali saattaa aiheuttaa ahdistusta ja turhautumista [Mirzakhani et al., 2010].

Sähköisen etäkurssin etuja on sen oppijakeskeisyys. Kurssin etenemistahti on kiinni opiskelijasta itsestään, mikäli kaikki materiaali kurssille on valmiina. Hitaammat oppijat voivat rauhassa keskittyä omaksumaan tietoa ja edetä itselleen sopivassa tahdissa. Sama vapaus määrittää tahtinsa pätee niihin opiskelijoihin, jotka haluavat edetä vauhdilla. Kun luennot ja harjoitukset eivät ole sidottuja tiettyyn aikaan, pystyvät opiskelijat määrittämään omaan aikatauluunsa parhaimmin sopivan hetken kurssin luentoja ja harjoituksia varten. Verkkopohjainen ratkaisu myös sallii opiskelijoiden osallistua kurssille, vaikka he eivät sillä hetkellä olisikaan samalla paikkakunnalla opettajan ja muiden kurssilaisten kanssa. [Mirzakhani et al., 2010]

Sähköisen etäkurssin materiaali on usein helpompi pitää ajan tasalla verrattuna esimerkiksi oppikirjoihin, sillä sähköisen materiaalin muokkaaminen on nopeaa. Tämä mahdollistaa myös kurssin materiaalien helpon uudelleenkäytön toisessa yhteydessä. Opiskelijoiden on mahdollista sähköisellä etäkurssilla kerrata tietoja materiaalista helpommin kuin esimerkiksi luentokurssilla, josta ei välttämättä ole sähköistä tallennetta. Perinteisellä luentokurssilla on vaikea selvittää ilman muistiinpanoja tai sähköistä tallennetta, mitä luennoitsija sanoi kolmannella luennolla, kun taas sähköisellä etäkurssilla tätä ongelmaa ei synny. [Mirzakhani et al., 2010]

Sähköisessä oppimisympäristössä ei myöskään tule ongelmaa opiskelijoiden määrästä. Perinteisellä luentokurssilla opiskelijoiden määrää rajaa luentosalin koko, kun taas verkossa opiskelijoiden määrälle ei ole yhtä selkeää fyysistä rajoitetta. Koska luentoja ei järjestetä samassa tilassa muiden kurssilaisten kanssa, todennäköisyys luennon häiriintymiselle selän takana supisevien kaverusten takia on hyvin pieni. [Mirzakhani et al., 2010]

Kuitenkin on hyvä myös muistaa, että jokaisella kolikolla on kääntöpuolensa. Sähköisiin oppimisympäristöihin ja niiden avulla toteutettuihin kursseihin liittyy haasteita, joihin vastaamalla voimme parantaa niiden käyttäjäkokemusta. Yksi selkein ongelma sähköisillä etäkursseilla verrattuna perinteisiin luentokursseihin on välittömän palautteen vähyys. Kasvokkaisessa vuorovaikutuksessa on nopea pyytää selitystä asiasta, jota ei käsittänyt. Perinteisellä tavalla toteutetuissa harjoitusryhmissä on myös mahdollista saada nopeasti palautetta omasta ratkaisustaan. Tämä ei ole yhtä helppoa etäkurssilla. [Mirzakhani et al., 2010]

Kasvokkaisen vuorovaikutuksen puute hidastaa myös reagointia opiskelijan ongelmiin sekä opittavan asian että tekniikan kanssa. Opiskelijalla on suurempi vastuu omasta oppimisestaan ja tarvittavien teknisten taitojen hankkimisesta [Petre et al., 1997]. Turhautumisen välttämiseksi on kuitenkin hyvä tarjota selkeät kanavat teknisen ja muun tuen saamiseksi. Tästä myös seuraa, että materiaalien valmisteluun menee enemmän aikaa, sillä niiden pitää olla kattavia julkaisuhetkellä. Perinteisellä luennolla luennoitsija voi sen sijaan korjata lennosta virheet luentokalvoissa, täydentää niiden sisältöä luennon aikana ja vastata tarkentaviin kysymyksiin.

PUXUI-kurssin käyttäjäkokemus muotoutui kyselyiden perusteella positiiviseksi. Ilmenneet ongelmat olivat pääasiassa teknisiä, joihin voidaan varautua tulevaisuudessa paremmalla ohjeistuksella, ja osa korjattiin jo kurssin aikana. Hyvinä asioina taas koettiin kurssin aihe, kurssin toteutustapaan liittyvät asiat ja materiaalien monipuolisuus.

Tulevaisuutta ajatellen PUXUI-kurssista löytyy muutamia hyviä kehityskohteita. Yksi paljon kommentteja saanut ongelmakohta oli harjoitusten kokeminen aikaa vieviksi. Mikäli kurssilaisille olisi selkeämmin näkyvissä etukäteen, paljonko aikaa harjoitusten tekeminen keskimäärin vie, eivät he välttämättä yrittäisi tehdä kaikkea yhden illan aikana. Tällöin urakka tuntuisi pienemmältä kuin tilanteessa, jossa kurssilainen käyttää harjoitusten tehtävien parissa koko päivän. Harjoitusten tehtävien yhteyteen olisi myös hyvä lisätä jokin indikaattori, joka kertoo, montako tehtävää kyseisessä harjoituksessa on vielä jäljellä. Kurssilaisen tietäisi tarkalleen, missä kohtaa harjoitusta on, eikä se tuntuisi jatkuvan ikuisesti.

Muutama tekninen ongelma johtui siitä, että kurssilaiset tekivät tehtäviä yliopiston verkon ulkopuolella tai hitaalla verkkoyhteydellä. Kurssin aluksi voisi olla hyvä mainita, että osa kurssin materiaalista on nopeaa verkkoyhteyttä vaativia videoita ja osa tieteellisiä tekstejä, joihin pääsee käsiksi helpoimmin yliopiston tietokoneluokkien koneilta. Videoita tai rajatun pääsyoikeuden omaavia tieteellisiä artikkeleja sisältävät harjoitukset voisi myös merkitä jonnekin näkyville, jotta kurssilaiset voisivat varautua niihin. Näin kurssilaiset tietäisivät varautua mahdollisiin ongelmiin kyseisten harjoitusten yhteydessä, mikäli päättäisivät tehdä niitä omilla koneillaan hitaalla verkkoyhteydellä.

Moni välikyselyyn vastannut kurssilainen kaipasi tietoa omasta menestyksestään kurssin aikana. Yksittäisen palautteen antaminen jokaisesta harjoituksesta olisi hidasta ja työlästä kurssin vetäjille, joten se ei ole varteenotettava vaihtoehto. Yksi tapa voisi olla järjestää kurssin puolivälissä yhtenä harjoituksena monivalintatestin, jonka tuloksen perusteella opiskelijat pystyisivät arvioimaan oman osaamisensa tasoa. Toinen vaihtoehto olisi liittää jokaiseen harjoitukseen yksi pohdintatehtävä, joka toteutettaisiin avoimena keskusteluna tehtävälle omistetulla foorumilla. Vuorovaikutuksessa muiden kurssilaisten kanssa kurssilaiset saisivat suoraa palautetta vertaisiltaan, mutta samalla he pystyisivät myös vertaamaan omaa tasoaan muiden tasoon.

Sähköiset oppimisympäristöt ja etäkurssit tulevat tuskin korvaamaan perinteistä opetusta ennen kuin kaikkiin niiden haasteisiin on löydetty tyydyttävä ratkaisu. Sähköisiä oppimisympäristöjä on kuitenkin perusteltua käyttää tukemaan perinteistä opetusta, jolloin molemmista tavoista voidaan saada suurin hyöty. Parhaimmassa tapauksessa tämä parantaisi opetuksen laatua ja avaisi osallistumismahdollisuuksia suuremmalle joukolle opiskelijoita.

Viiteluettelo

[Karapanos et al., 2009] E. Karapanos, J. Zimmerman, J. Forlizzi and J.-B. Martens, User experience over time: an initial framework. In: *Proc. of CHI 2009*, 729-738.

[Konstantinidis *et al.*, 2009] A. Konstantinidis, Th. Tsiatsos and A. Pomportsis, Collaborative virtual learning environments: design and evaluation. *Multimedia Tools and Applications* **44**, 2 (2009), 279-304.

[Lin, 2012] W.-S. Lin, Perceived fit and satisfaction on web learning performance: IS continuance intention and task-technology fit perspectives. *International Journal of Human-Computer Studies* **70**, 7 (2012), 498-507.

[Law and Sun, 2012] E. L.-C. Law and X. Sun, Evaluating user experience of adaptive digital educational games with Activity Theory. *International Journal of Human-Computer Studies* **70**, 7 (2012), 478–497.

[Mirzakhani *et al.*, 2010] M. Mirzakhani, H. Ashrafzadeh and A. Ashrafzadeh, The virtual university: Advantages and disadvantages. In: *Proc. of the 4th International Conference on Distance Learning and Education*, 32-36.

[Notess, 2001] M. Notess, Usability, user experience, and learner experience. e*Learn* **2001**, 8 (2001), 3.

[Petre et al., 1997] M. Petre, L. Carswell, B. Price and P. Thomas, Innovations in large-scale supported distance teaching: transformation for the Internet, not just translation. In: *Proc. of 27th Frontiers in Education Annual Conference* **1** (1999), 332-338.

[Richards and Kelaiah, 2012] D. Richards and I. Kelaiah, Usability attributes in virtual learning environments. In: *Proc. of 8th Australasian Conference on Interactive Entertainment* **2012**, article No. 9.

Itseohjautuvien järjestelmien arkkitehtuurit

Miia Ketolainen

Tiivistelmä.
Tässä tutkielmassa käsitellään itseohjautuvien järjestelmien arkkitehtuureja sekä niihin liittyviä suunnittelumalleja. Tutkielman painopiste on itseohjautuvien järjestelmien rakenteessa ja käyttäytymisessä. Lisäksi tarkastellaan mekanismeja, joiden avulla järjestelmät voivat itsenäisesti päättää, miten niiden tulisi mukautua.
Avainsanat ja -sanonnat: itseohjautuvat järjestelmät, ohjelmistoarkkitehtuurit, suunnittelumallit
CR-luokat: D.2.11

1. Johdanto

Itseohjautuvilla järjestelmillä tarkoitetaan järjestelmiä, jotka pystyvät mukautumaan ohjelmassa ja sen suoritusympäristössä tapahtuviin muutoksiin ajon aikana. Tällaisia muutoksia voivat olla esimerkiksi käyttäjän antama syöte tai ulkoisista laitteista tai sensoreista tuleva informaatio. Sellaiset muutokset, joihin järjestelmä ei ole varautunut, aiheuttavat lisäkustannuksia virheiden etsimisen, korjaamisen ja järjestelmän uudelleenasentamisen muodossa. Järjestelmän itseohjautuvuus vähentäisi näitä kustannuksia ja samalla säästäisi aikaa.

Itseohjautuvuutta suunnitellessa on otettava huomioon järjestelmän tavoitteet ja vaatimukset, järjestelmän itseohjautuvaan käyttäytymiseen johtavat tekijät, mekanismit, joilla itseohjautuvuus saavutetaan sekä itseohjautuvuuden vaikutukset järjestelmään [Cheng et al., 2009].

Itseohjautuvissa järjestelmissä on usein kaksi tai useampia alijärjestelmiä, joista tässä työssä käytetään termejä ohjaava alijärjestelmä ja ohjattava alijärjestelmä. Ohjattavissa järjestelmissä on koko järjestelmän toiminnallisuus, kun taas ohjaavien järjestelmien tehtävänä on tarkkailla ympäristöä sekä ohjattavia järjestelmiä, analysoida mukautumisen tarve, suunnitella mukautumisprosessi ja lopulta antaa ohjattavalle järjestelmälle käsky mukautua.

Luvussa 2 käsitellään itseohjautuvia järjestelmiä ja niiden ominaisuuksia ja suunnittelua. Luvussa 3 esittelen arkkitehtuurityylejä ja luvussa 4 suunnittelumalleja, jotka tukevat järjestelmien itseohjautuvien ominaisuuksien suunnittelua. Luku 5 sisältää yhteenvedon aiheesta sekä ajatuksia siitä, miten aihetta tulisi jatkotutkimuksissa käsitellä.

2. Itseohjautuvat järjestelmät

Tässä luvussa käsittelen itseohjautuvia järjestelmiä sekä niiden ominaisuuksia ja suunnittelua. Kohdassa 2.1 esittelen järjestelmien ominaisuuksia ja kohdassa 2.2 käsittelen itseohjautuvien järjestelmien suunnittelua ja suunnittelun lähtökohtia.

2.1. Itseohjautuvien järjestelmien ominaisuudet

Itseohjautuviksi järjestelmiksi kutsutaan sellaisia järjestelmiä, jotka pystyvät mukautumaan ympäristössään tai järjestelmässä itsessään tapahtuviin muutoksiin ilman että järjestelmän toiminta häiriintyy. Salehie ja Tahvildari [2009] esittävät itseohjautuvuuden hierarkkisena käsitteenä, joka koostuu kolmesta tasosta: yleisestä tasosta, päätasosta ja primitiivisestä tasosta. Yleisen tason ominaisuuksia ovat itseohjautuvuus ja itseorganisoituvuus. Itseohjautuvuuden käsitteen alle luetaan kuuluvaksi sellaiset itse-etuliitteelliset ominaisuudet kuin *itsejohtavuus* (self-managing), *itsehallitsevuus* (self-governing), *itseylläpito* (self-maintenance), *itsevalvonta* (self-control) ja *itsearviointi* (self-evaluating). Itseorganisoituviksi sanotaan järjestelmiä, joissa hajauttamisella ja uusilla toiminnoilla on suuri merkitys.

Päätasolle kuuluu Salehien ja Tahvildarin [2009] mukaan neljä ominaisuutta, *itse-konfiguroituvuus* (self-configuring), *itsekorjautuvuus* (self-healing), *itseoptimoituvuus* (self-optimizing) ja *itsesuojautuvuus* (self-protecting). Itsekonfiguroituvuudella tarkoitetaan järjestelmän kykyä uudelleenkonfiguroitua automaattisesti asentamalla, päivittämällä ja yhdistämällä ohjelmiston osia. Itsekorjautuvuus määritellään järjestelmän kyvyksi huomata, tunnistaa ja reagoida itsessään tapahtuviin virheisiin ja mukautua siten, että järjestelmä pysyy toimintakykyisenä. Itseoptimoituvuus tarkoittaa järjestelmän kykyä hallita suorituskykyään ja resurssien käyttöä täyttääkseen käyttäjien vaatimukset. Itsesuojautuvuus on järjestelmän kyky havaita tietoturvauhkia ja toipua niiden vaikutuksista.

Primitiivisellä tasolla ovat sellaiset ominaisuudet kuin *itsetietoisuus* (self-awareness), *itsetarkkaileminen* (self-monitoring), *itsesijainti* (self-situated) ja *kontekstitietoisuus* (context-awareness). Itsetietoisuudella tarkoitetaan sitä, että järjestelmä on tietoinen omasta tilastaan ja käyttäytymisestään. Tämä ominaisuus liittyy hyvin vahvasti itsetarkkailemiseen, joka määritellään järjestelmän kyvyksi tarkkailla toimintaansa. Kontekstitietoisuus tarkoittaa järjestelmän kykyä olla tietoinen ympäristönsä tilasta.

Eri tutkijoilla on erilaisia näkemyksiä siitä, mitkä ominaisuudet kuuluvat minkäkin käsitteen alle. Esimerkiksi Mühlin ja muiden [2007] mukaan itsejohtavuus on itseorganisoituvuuden yläkäsite. He määrittelevät käsitteiden hierar-

kian joukko-opin avulla siten, että *itsejohdettavat* (self-manageable) järjestelmät ovat itseohjautuvien järjestelmien osajoukko, jonka osajoukko puolestaan ovat *itsejohtavat* järjestelmät (self-managing). Itsejohtavia ovat sellaiset järjestelmät, joilla on keskitetty ohjauskomponentti, jonka poistamisesta seuraa, että järjestelmä menettää mukautumiskykynsä. *Itseorganisoituvat* järjestelmät (self-organizing) puolestaan ovat itsejohtavien järjestelmien sellainen osajoukko, joilla ei ole keskitettyä ohjauskomponenttia.

Tässä tutkielmassa itseohjautuvuus kuitenkin käsittää kaikki edellä mainitut ominaisuudet. Ollakseen itseohjautuva järjestelmän tulee kyetä tarkkailemaan itseään tai ympäristöään tai molempia. Tämä ei kuitenkaan vielä riitä, vaan järjestelmän täytyy lisäksi kyetä muuttamaan käyttäytymistään tai rakennettaan oman tai ympäristönsä tilan mukaan. Itseohjautuvalla järjestelmällä on siis jompikumpi tai molemmat ominaisuuksista itsetarkkaileminen ja ympäristön tarkkaileminen. Mikäli järjestelmä on itsetarkkaileva, seuraa siitä, että järjestelmä on itsetietoinen. Vastaavasti kyvystä tarkkailla ympäristöä seuraa ominaisuus kontekstitietoinen.

Ollakseen itsekorjautuva, järjestelmällä tulee olla neljä ominaisuutta: vian ennustaminen ja havaitseminen, virheen diagnosoiminen tai paikallistaminen, virheen eristäminen tai viasta toipuminen ja validointi. [Gorla, 2007]

Korjausmekanismit käynnistyvät, kun järjestelmässä tapahtuu jokin virhe. Jotta mekanismit alkavat toimia, järjestelmän täytyy joko kyetä ennustamaan vian tapahtuminen tai havaitsemaan tapahtunut virhe, jotta se voi mukautua siten, että se pysyy toimintakykyisenä. Pystyäkseen huomaamaan tapahtumassa olevan tai jo tapahtuneen virheen järjestelmän täytyy pystyä havaitsemaan poikkeavuudet käyttäytymisessään. Sen tulee siis havainnoida itseään ja tunnistaa normaalista poikkeava käytös keräämässään informaatiossa. [Gorla, 2007]

Jotta virheistä toipuminen olisi mahdollista, järjestelmän täytyy pystyä paikallistamaan virheen aiheuttaja. Paikallistamisen ei välttämättä tarvitse onnistua kovin suurella tarkkuudella, vaan järjestelmä voi esimerkiksi tunnistaa komponentin, jossa virhe tapahtui. Järjestelmän tulee siis olla tietoinen omasta tilastaan sekä komponenttien ja niitä pienempien ohjelmistoyksiköiden tiloista. [Gorla, 2007]

Diagnosoituaan vian tai paikallistettuaan virheen aiheuttajan korjausmekanismit voivat joko yrittää poistaa virheen tai eristää sen niin, että se ei pysty aiheuttamaan vahinkoa. Voidakseen eristää vian aiheuttajan ja toipua sen seurauksista järjestelmän pitää pystyä muokkaamaan suoritustaan muuttamalla joko tilojen järjestystä tai lauseiden suoritusjärjestystä tai molempia. Muutokset

voivat olla joko pysyviä tai väliaikaisia ja ne voivat tapahtua eri tasoilla, kuten komponenttitasolla tai metoditasolla. [Gorla, 2007]

Kun vian aiheuttaja on poistettu, järjestelmän täytyy varmistaa, että korjaustoimenpiteiden jälkeen järjestelmä täyttää edelleen sille asetetut toiminnalliset ja ei-toiminnalliset vaatimukset. Järjestelmän tulee siis olla myös itsearvioituva. [Gorla, 2007]

Itseorganisoituviksi sanotaan sellaisia järjestelmiä, jotka toimivat ilman keskitettyä ohjauskomponenttia. Sen sijaan niiden toiminta perustuu komponenttien väliseen paikalliseen vuorovaikutukseen. Jokainen komponentti suorittaa omaa tehtäväänsä itsenäisesti ja itseorganisoituvuus syntyy komponenttien välisestä vuorovaikutuksesta. [Serugendo et al., 2004]

2.2. Itseohjautuvien järjestelmien suunnittelu

Anderson ja muut [2009] käsittelevät itseohjautuvien järjestelmien mallintamista. He jakavat mallintamisen neljään eri osa-alueeseen: järjestelmän tavoitteet, muutos, mekanismit ja vaikutukset. Tavoitteilla tarkoitetaan koko järjestelmän tavoitteita, muutoksella tarkoitetaan niitä syitä, jotka aiheuttavat järjestelmän mukautumisen, mekanismeilla tarkoitetaan sitä, kuinka järjestelmä reagoi muutoksiin ja vaikutuksilla sitä, miten mukautuminen vaikuttaa järjestelmään.

Järjestelmän tavoitteita mallinnettaessa on tärkeää ottaa huomioon viisi eri näkökulmaa: evoluutio, joustavuus, kesto, moninaisuus ja riippuvuus. Evoluutionäkökulmalla tarkoitetaan sitä, että on mietittävä, voivatko järjestelmän tavoitteet muuttua sen elinkaaren aikana. Järjestelmän kehittyessä sen tavoitteet saattavat muuttua, niitä voi tulla lisää tai ne voivat poistua. Joustavuusnäkökulma tarkoittaa sitä, miten joustavasti tavoitteet on määritelty, kuinka paljon niihin liittyy epävarmuutta ja ovatko ne tarkasti määriteltyjä, rajoitettuja vai täysin rajoittamattomia. Kestonäkökulmalla tarkoitetaan sitä, onko tavoite validi järjestelmän koko elinkaaren ajan. Tavoite voi olla joko väliaikainen tai pysyvä. Moninaisuusnäkökulma liittyy siihen, kuinka paljon järjestelmällä on itseohjautuvuuteen liittyviä tavoitteita. Niitä voi olla yksi tai useita. Riippuvuusnäkökulmalla tarkoitetaan sitä, että on otettava huomioon mahdolliset suhteet tavoitteiden välillä. Tavoitteet voivat vaikuttaa toisiinsa tai ne voivat olla riippumattomia toisistaan. Toisistaan riippuvat tavoitteet voivat joko täydentää toisiaan tai olla ristiriidassa keskenään. [Anderson et al., 2009]

Järjestelmän muutokseen liittyy neljä eri näkökulmaa: muutoksen lähde, tyyppi, esiintymistiheys ja ennakointi. Muutoksen lähde voi olla joko ulkoinen tai sisäinen. Lähteen ollessa järjestelmän sisäinen on otettava huomioon muutoksen lähteen sijainti: tapahtuuko muutos sovelluksessa itsessään, väliohjelmistossa vai infrastruktuurissa. Muutoksen tyyppi voi olla toiminnallinen, ei-

toiminnallinen tai tekninen. Toiminnallinen muutos tarkoittaa sitä, että järjestelmän tarkoitus muuttuu ja sen tuottamien palveluiden täytyy mukautua uuteen tarkoitukseen. Ei-toiminnallinen muutos tarkoittaa, että järjestelmän laatuominaisuus muuttuu. Tekninen muutos tarkoittaa joko ohjelmistossa tai laitteistossa tapahtuvaa muutosta, esimerkiksi väliohjelmiston päivitystä uuteen versioon. Esiintymistaajuusnäkökulmalla tarkoitetaan sitä, kuinka usein tai harvoin muutos tapahtuu. Ennakointinäkökulma tarkoittaa sitä, voidaanko muutos ennustaa etukäteen ja voiko siihen varautua. Muutos voi olla joko odottamaton, ennustettavissa tai odotettavissa. Odottamattomaan muutokseen ei voida varautua, ennustettavissa olevaan muutokseen voidaan varautua ja odotettavissa oleva muutos voidaan ottaa huomioon järjestelmän suunnittelussa. [Anderson et al., 2009]

Itseohjautuvan järjestelmän mekanismien suunnitteluun liittyy seitsemän eri näkökulmaa: mekanismin tyyppi, autonomia, organisointi, laajuus, kesto, ajantasaisuus ja käynnistys. Mekanismin tyypillä tarkoitetaan sitä, liittyykö mukautuminen järjestelmän komponentin parametreihin vai järjestelmän rakenteeseen. Mukautumismekanismi voi siis olla joko rakenteellinen tai parametrillinen. Se voi olla myös näiden kahden tyypin yhdistelmä. Autonomianäkökulmalla tarkoitetaan sitä, liittyykö mukautumismekanismiin ulkopuolisia tekijöitä. Jos mekanismi on täysin autonominen, järjestelmän ulkopuoliset tekijät eivät vaikuta siihen, miten sen täytyy mukautua. Sen sijaan mekanismi on avustettu, jos jokin ulkopuolinen taho, kuten ihminen tai toinen järjestelmä, vaikuttaa siihen. Organisointinäkökulmassa on kyse siitä, onko mukautumismekanismi keskitetty vai hajautettu. Jos mekanismi on keskitetty, siitä on vastuussa yksi komponentti. Jos se on hajautettu, siihen osallistuu useita eri komponentteja. Laajuusnäkökulma liittyy siihen, koskeeko mukautuminen tiettyä osaa järjestelmästä vai koko järjestelmää. Mukautuminen voi tapahtua joko paikallisesti, jolloin on kyse järjestelmän osasta, tai globaalisti, jolloin on kyse koko järjestelmän laajuisesta mekanismista. Kestonäkökulmassa on kyse siitä, kuinka kauan mukautumisprosessi kestää. Prosessin pituus voi olla lyhyt, keskipitkä tai pitkä, tosin luokittelu on aina riippuvainen järjestelmän sovellusalueesta. Se, mikä yhdessä järjestelmässä on lyhyt aika, voi toisessa olla pitkä. Ajantasaisuusnäkökulmalla tarkoitetaan sitä, miten hyvin voidaan taata mukautumisen tapahtuvan tietyssä ajanjaksossa. Mikäli muutos tapahtuu usein, ei välttämättä voida taata, että edellisen muutoksen laukaisema sopeutumismekanismi on suoritettu loppuun ennen kuin seuraava muutos tapahtuu. Käynnistysnäkökulma tarkoittaa sitä, onko mukautumismekanismin laukaiseva muutos aika- vai tapahtumaperustainen. [Anderson et al., 2009]

Vaikutuksiin liittyviä näkökulmia on neljä: kriittisyys, ennustettavuus, negatiivisuus ja sietokyky. Kriittisyysnäkökulma liittyy siihen, miten mukautumisen epäonnistuminen vaikuttaa järjestelmään. Jotkut vaikutukset voivat olla harmittomia, mutta epäonnistumisesta voi seurata myös kriittisiä ongelmia. Ennustettavuusnäkökulmassa on kyse siitä, missä määrin mukautumisen vaikutuksia voidaan ennustaa. Ennustettavuutta voidaan luonnehtia deterministiseksi tai epädeterministiseksi. Deterministisellä tarkoitetaan sitä, että vaikutukset tiedetään melko tarkasti etukäteen, kun taas epädeterminististen tarkoittaa sitä, että vaikutuksia on hankala ennustaa tarkasti. Negatiivisuusnäkökulmalla tarkoitetaan sitä, millaisia negatiivisia vaikutuksia mukautumisella on järjestelmän suorituskykyyn. Vaikutukset voivat olla merkityksettömiä tai pahimmassa tapauksessa se voi johtaa järjestelmän kaatumiseen. Sietokykynäkökulmassa on kyse siitä, miten hyvin järjestelmä pystyy toimittamaan tarjoamansa palvelut muutosten tapahtuessa. [Anderson et al., 2009]

3. Arkkitehtuurityylit

Tässä luvussa käsittelen eri arkkitehtuurityylejä. Arkkitehtuurityylillä tarkoitetaan yleistä mallia, jonka mukaan järjestelmä organisoidaan ylimmällä abstraktiotasolla ja joka määrää järjestelmän teknisen luonteen. Yleisesti tunnettuja arkkitehtuurityylejä ovat kerrosarkkitehtuurit, viestinvälitysarkkitehtuurit, tietovuoarkkitehtuurit, asiakas-palvelin-arkkitehtuurit, MVC-arkkitehtuurit ja tulkkiarkkitehtuurit.

Oreizy ja muut [2008] käsittelevät arkkitehtuurityylejä, jotka tukevat itseohjautuvuutta. Tällaisia tyylejä ovat Pipe and Filter, Weaves, Publish-Subscribe, C2, Tile Style, Representational State Transfer (REST), Computational REST (CREST) ja MapReduce. Tässä luvussa esitellään ensin lyhyesti edellä mainittuja arkkitehtuurityylejä. Lisäksi kohdassa 3.1 käsitellään tarkemmin C2-arkkitehtuuria ja kohdassa 3.2 siihen perustuvaa PitM-arkkitehtuuria, sillä ne soveltuvat hyvin lähtökohdaksi monenlaisiin itseohjautuviin järjestelmiin. Muiden käyttö rajoittuu tietynlaisten ongelmien ratkaisemiseen.

Pipe and Filter -tyylissä olennaista on erilliset komponentit sekä komponenttien väliset yhdistäjät, rajapintojen käyttö ja standardimuotoiset viestit. Arkkitehtuuri koostuu suodatinkomponenteista (filter), jotka saavat sisäänsä tietoa, käsittelevät sen, ja siirtävät väylää (pipe) pitkin eteenpäin seuraavalle suodatinkomponentille. Uuden järjestelmän luominen olemassa olevista suodattimista ja väylistä on yksinkertaista, mutta tyyli ei kuitenkaan tue ajonaikaista muutosta. Esimerkiksi ajonaikainen uudelleenreititys yhdestä komponentista toiseen ei onnistu. [Oreizy et al., 2008]

Weaves-tyyli on kehittynyt tietovuoarkkitehtuurin pohjalta ja se mahdollistaa ajonaikaiset muutokset yhdistämällä viestien puskuroinnin ja tietovoiden hallintamekanismit. Kun väylä irrotetaan sen jälkeisestä suodattimesta, väylän puskuri täyttyy viesteistä. Kun puskurin kapasiteetti on melkein täynnä, väylä pyytää sitä edeltävän suodattimen rajapintaa hidastamaan viestien tuotantoa siihen asti kunnes uuden komponentin liittäminen on suoritettu loppuun ja puskuri pystyy taas käsittelemään uusia viestejä. [Oreizy et al., 2008]

Publish-subscribe -arkkitehtuuri on viestinvälitysarkkitehtuuri, jossa julkaisijan tehtävänä on lähettää viestejä ja tilaajan tehtävänä on vastaanottaa niitä. Tilaaja rekisteröityy vastaanottajaksi ja julkaisija pitää kirjaa vastaanottajista. [Oreizy et al., 2008]

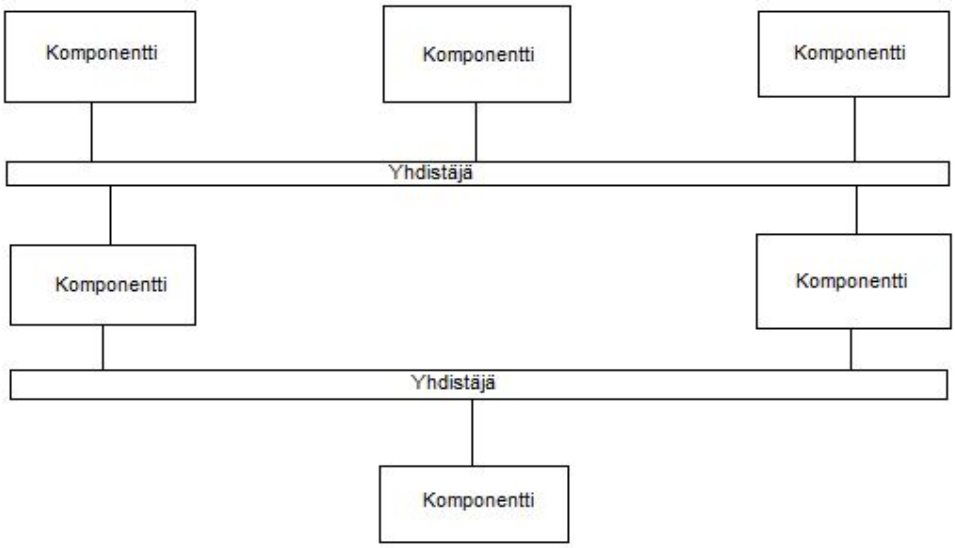
Tile Style -arkkitehtuurissa hyödynnetään tietokoneverkoston laskentavoimaa tarkoituksena ratkaista NP-täydellisiä ongelmia ja sen taustalla on malli molekyylien välisistä sidoksista. Arkkitehtuurin etuja ovat tietoturva, vakaus ja skaalautuvuus. Arkkitehtuurin lähtökohtana on järjestelmä, jolla on todistetusti pystytty ratkaisemaan jokin tunnettu NP-täydellinen ongelma. Jokaista kyseisessä järjestelmässä olevaa laattaa kohti vastaava Tile Style -arkkitehtuuri ottaa käyttöön yksinkertaisen laattakomponentin yhdessä verkostoon kuuluvista tietokoneista, yhdistää kyseisen komponentin muihin verkoston komponentteihin halutun arkkitehtuurin mukaisesti ja sitten luo kaksi kopiota komponentista. Sama prosessi toistetaan molemmille kopioille niin kauan, että ongelma saadaan ratkaistua tai kunnes on riittävän todennäköistä, että ratkaisua ei ole olemassa. Laattojen kahdentaminen johtaa siihen, että haavoittuvuudet ja yksittäisten komponenttien kaatumiset eivät estä järjestelmän toimintaa. [Oreizy et al., 2008]

MapReduce-arkkitehtuuria voidaan käyttää suurten tietomäärien prosessoimiseen. Siinä määritellään kaksi toimintoa, joista toinen jakaa tietomäärän pienempiin osiin, jotka sitten käsitellään rinnakkaisesti, ja toinen sulauttaa käsitellyn tiedon takaisin yhteen. Mikäli joku tietoa käsittelevistä yksiköistä kaatuu, sen käsittelemä tieto ohjataan automaattisesti jollekin toimivalle yksikölle. [Oreizy et al., 2008]

REST-tyylin tunnetuin toteutus lienee Internet. Se muuttuu jatkuvasti palvelimien, asiakkaiden, porttien ja välittäjien vaihtuessa. Se on suunniteltu hypermediadokumenttien jakamiseen. REST:ssä tiedon siirtäminen perustuu resursseihin, joihin viitataan URL:illa. Resurssit esitetään tavuina sekä tavuja kuvaavana metatietona. CREST sen sijaan perustuu tiedon siirtämisen sijaan tiedon käsittelyn siirtämiseen. Tiedon käsittely voi olla esimerkiksi tekstin käsittelyä tai kuvan muokkaamista. CREST:ssä resurssi on ohjelma sekä sitä kuvaava metatieto. [Oreizy et al., 2008]

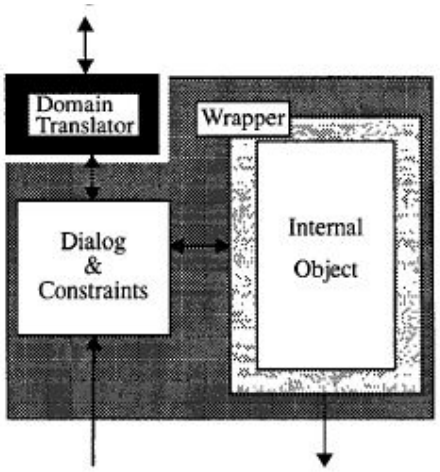
3.1. C2-arkkitehtuuri

C2-arkkitehtuuri on eräänlainen yhdistelmä viestinvälitysarkkitehtuurista ja kerrosarkkitehtuurista. Se koostuu komponenteista ja yhdistäjistä, joilla on kaksi porttia, yksi yläportti ja yksi alaportti, joiden kautta komponentit lähettävät viestejä toisilleen. Komponentin yläportti voidaan liittää yhdistäjän alaporttiin ja komponentin alaportti yhdistäjän yläporttiin. Yhteen yhdistäjään voidaan liittää useita komponentteja ja toisia yhdistäjiä. Kuvassa 1 on yksinkertainen C2-arkkitehtuuri.



Kuva 1. C2-arkkitehtuuri. [Taylor et al., 1995]

Komponenttien sisäinen rakenne on esitetty kuvassa 2. Komponentilla on sisäinen olio, joka tarjoaa rajapinnan. Olion ympärillä on ns. kääre. Kääre toimii siten, että kun jotakin olion rajapinnan metodia kutsutaan, kääre muodostaa kutsusta ja sen paluuarvosta ilmoituksen, jonka se välittää yhdistäjällä, joka on liitetty komponentin alaporttiin. Lisäksi komponentti sisältää ohjelman, joka ylläpitää vuoropuhelua ja rajoitteita. Komponentti voi sisältää myös osakomponentin, jonka tehtävänä on kääntää komponentin yläporttiin liitetyltä yhdistäjältä tulevat viestin komponentin ymmärtämään muotoon. [Taylor et al., 1995]



Kuva 2. Komponentin rakenne C2-arkkitehtuurissa [Taylor et al., 1995].

Komponentit voivat lähettää kahdenlaisia viestejä, ilmoituksia ja pyyntöjä. Ilmoitukset kulkevat arkkitehtuurissa alaspäin ja pyynnöt ylöspäin. Ilmoituksilla komponentit lähettävät toisilleen tiedon niiden sisäisten olioiden tiloissa tapahtuneista muutoksista. Pyyntöjen avulla komponentit sen sijaan ohjeistavat yläpuolellaan olevia komponentteja toimimaan halutulla tavalla. Ilmoituksilla viestitään siis sitä, mitä toimenpiteitä on tehty, millä parametreilla ja mitä paluuarvoja on saatu, ja pyynnöillä viestitään sitä, mitä toimenpiteitä halutaan tehdä. [Taylor et al., 1995]

Yhdistäjien tehtävänä on liittää komponentit toisiinsa sekä reitittää ja välittää viestejä. Ilmoitukset voidaan välittää joko kaikille yhdistäjän alaporttiin liitetyille komponenteille ja pyynnöt vastaavasti kaikille yläporttiin liitetyille komponenteille, tai ne voidaan välittää vain sellaisille komponenteille, jotka ovat tilanneet kyseisentyyppiset viestit. Viestejä voidaan välittää myös siten, että ne lähetetään komponenteille tietyssä järjestyksessä niin kauan kunnes jokin määrätty ehto täyttyy. [Taylor et al., 1995]

3.2. PitM-arkkitehtuuri

PitM-arkkitehtuuri perustuu C2-arkkitehtuuriin. Se koostuu komponenteista ja yhdistäjistä, jotka kommunikoivat välittämällä viestejä. C2-arkkitehtuurin ylä- ja alaporttien lisäksi komponenteilla voi olla myös sivuportteja.

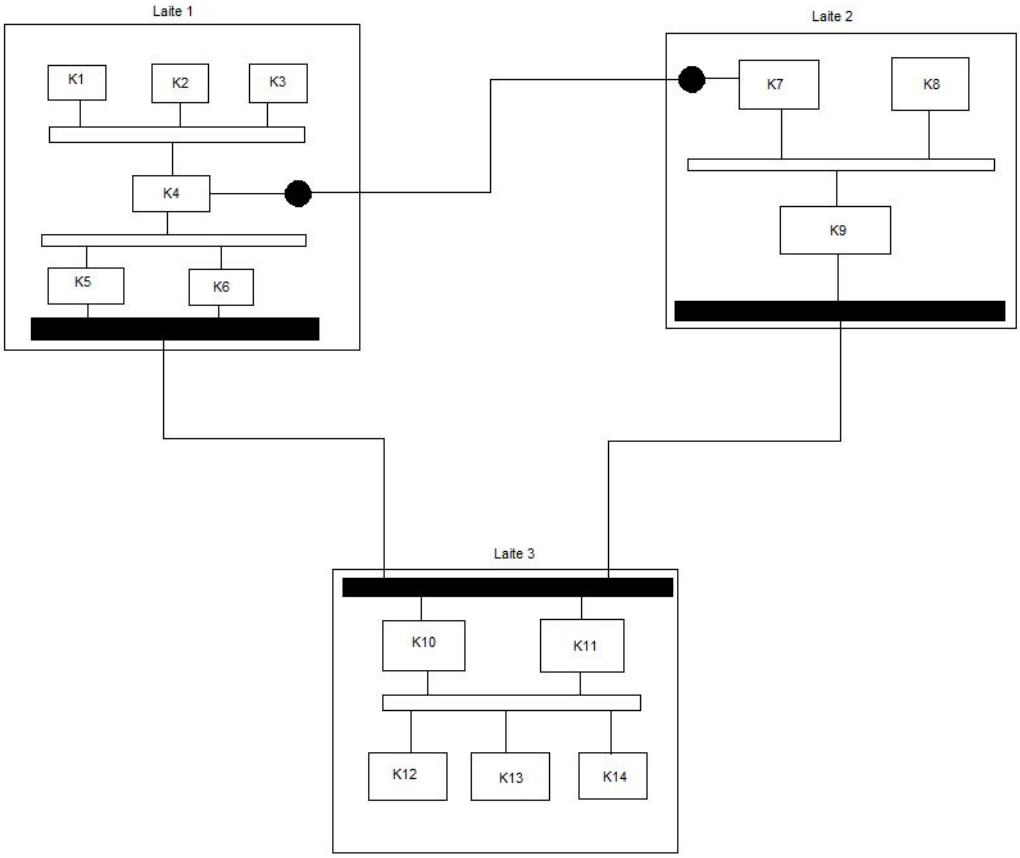
Ilmoitusten ja pyyntöjen lisäksi PitM-tyylissä on myös kolmas viestityyppi, *vertaisviestit* (peer). Lisäksi PitM-arkkitehtuurissa voi olla myös ns. *reunayhdistäjiä* (border connector), jotka toimivat eri laitteissa sijaitsevien komponenttien välissä. [Medvidovic and Mikic-Rakic, 2001]

Komponenttien sivuportteihin liitetään ns. *vertaisyhdistäjiä* (peer connector), joiden kautta komponentit voivat lähettää toisilleen vertaisviestejä. Kun viesti tulee vertaisyhdistäjään, se välitetään eteenpäin muiden porttien kautta niille

komponenteille, jotka on liitetty portteihin. Viestit voidaan välittää myös määrättyjen ehtojen mukaisesti, samoin kuin C2-tyylissä. [Medvidovic and Mikic-Rakic, 2001]

PitM-arkkitehtuurissa komponentit eivät voi lähettää toisilleen vertaisviestejä, mikäli niiden välillä on vertikaalinen yhteys. Myöskään vertaisyhdistäjien ja tavallisten yhdistäjien väliset viestit eivät ole sallittuja, sillä silloin vertaisviestit pitäisi muuttaa pyynnöiksi ja ilmoituksiksi ja päinvastoin. [Medvidovic and Mikic-Rakic, 2001]

Kuvassa 3 on esitetty PitM-arkkitehtuurin rakenne järjestelmässä, jossa on kolme laitetta. Mustat suorakaiteet kuvaavat reunayhdistäjiä ja mustat ympyrät vertaisyhdistäjiä. Laitteet 1 ja 2 ovat laitteen 3 yläpuolella, joten vertaisyhdistäjät niiden komponenttien ja laitteen 3 komponenttien välillä eivät ole sallittuja. Laitteiden 1 ja 2 välillä sen sijaan ei ole vertikaalista yhteyttä, joten niiden sisältämät komponentit voivat lähettää vertaisviestejä toisilleen vertaisyhdistäjien kautta.



Kuva 3. PitM-arkkitehtuuri [Medvidovic and Mikic-Rakic, 2001]

4. Suunnittelumallit

Suunnittelumallit ovat abstrakteja uudelleenkäytettäviä ratkaisuja usein esiintyviin ongelmiin. Gamman ja muiden [1995] mukaan suunnittelumallit koostuvat neljästä elementistä: nimestä, ongelmasta, ratkaisusta ja seurauksista. Jokaisella mallilla on nimi, jota käytetään kuvaamaan kolmea muuta elementtiä. Ongelma kuvaa tilanteen, jossa suunnittelumallia käytetään. Ratkaisu kuvaa ne elementit, joita tarvitaan ongelman korjaamiseksi, ja seuraukset kuvaavat sen tilan, joka syntyy kun mallia on sovellettu ongelman ratkaisemiseksi.

Gomaa ja Hussein [2004] esittelevät suunnittelumalleja, joita voi käyttää tilanteissa, joissa järjestelmän täytyy muuttaa rakennettaan ajonaikaisesti. Tällaiset mallit määrittelevät, miten järjestelmän komponentit toimivat yhdessä muuttaakseen järjestelmän konfiguraatiota.

Myös Ramirez ja Cheng [2010] esittelevät erilaisia suunnittelumalleja itseohjautuvien järjestelmien suunnittelun avuksi. He ovat tutkineet kymmeniä itseohjautuvia järjestelmiä käsitteleviä artikkeleita ja projekteja, joista he ovat tunnistaneet erilaisia itseohjautuvuutta tukevia suunnittelumalleja. Näitä malleja he ovat soveltaneet itseohjautuvan uutispalvelun suunnittelussa.

Ramirez ja Cheng [2010] jakavat löytämänsä suunnittelumallit kolmeen kategoriaan, tarkkailumalleihin, päätöksentekomalleihin ja uudelleenkonfiguroitumista tukeviin malleihin. Tarkkailumalleja on kolme: Sensor Factory, Reflective Monitoring ja Content-based Routing. Päätöksentekomalleja ovat Adaptation Detector, Case-based Reasoning, Divide and Conquer, Arhitecture-based ja Tradeoff-based. Uudelleenkonfiguroitumista tukevia malleja ovat Component Insertion, Component Removal, Server Reconfiguration ja Decentralized Reconfiguration.

Sensor Factory -mallin tarkoitus on ottaa käyttöön sensoreita, jotka keräävät tietoa järjestelmän komponenteilta. Valvottavien komponenttien täytyy toteuttaa AbstractSensor-rajapinta, jonka kautta niiltä voidaan kysyä tarvittavaa tietoa. SensorFactory-luokka hallitsee sensorikokoelmaa siten, että sensorit eivät ole suoraan yhteydessä komponentteihin ja asiakkaisiin. Lisäksi Registry-luokka määrittelee, onko sensorien keräämä data jaettavaa vai ei. Malli mahdollistaa uusien sensorien liittämisen järjestelmään ja saattaa vähentää resurssien käyttöä jakamalla sensorien keräämää informaatiota, mutta saattaa samalla heikentää suorituskykyä sensorien ja asiakkaan väliin tulevan ylimääräisen kerroksen vuoksi. [Ramirez and Cheng, 2010]

Reflective Monitoring -mallissa komponenttiin liitetään mekanismi, jonka avulla se voi sekä tarkkailla omaa sisäistä tilaansa että tarjota sensoreille rajapinnan, jonka kautta ne voivat pyytää siltä informaatiota. Malli hyödyntää reflektiota kiertääkseen komponentin kapseloinnin ja tarjotakseen metodeja, joi-

den avulla voidaan päästä käsiksi komponentin attribuuttien arvoihin. Mallin avulla voidaan erottaa toisistaan tarkkailun mahdollistava rajapinta ja komponentin toiminnallinen logiikka. [Ramirez and Cheng, 2010]

Content-based Routing -malli tarjoaa ratkaisun tilanteeseen, jossa usea asiakaskomponentti tarvitsee samaa informaatiota, mikä vaikuttaa kyseistä informaatiota tarjoavan komponentin toimintaan. Sen sijaan, että asiakaskomponentit pyytäisivät tietoa suoraan sensoreilta, käytetään monta-moneen -tyyppistä tilaaja-tuottaja -mallia, joka ottaa tiedon sensorilta ja jakaa sen asiakaskomponenteille. Asiakaskomponentit määrittelevät, minkälaisista tapahtumista ne ovat kiinnostuneita, ja kun tilaaja-tuottaja -malli havaitsee sensoreiden tuottavan kyseisenlaista dataa, se toimittaa tiedon asiakaskomponenteille. [Ramirez and Cheng, 2010]

Adaptation Detector -mallissa sensoreilta saatavat tietovirrat liitetään arvoihin, jotka havainnollistavat järjestelmän odotetun ja havaitun käyttäytymisen eroavaisuutta. Jos poikkeama on tarpeeksi suuri, käynnistetään päätöksentekoprosessi, jossa ensin valitaan sopiva uudelleenkonfiguroitumistapa. Mallin avulla voidaan erottaa tarkkailuprosessi ja päätöksentekoprosessi toisistaan. [Ramirez and Cheng, 2010]

Case-based Reasoning -mallissa yhdistetään ennalta määritellyt tarkkailun tulokset tiettyihin uudelleenkonfiguroitumisohjeisiin. Malli soveltuu käytettäväksi tilanteissa, joissa erilaiset uudelleenkonfiguroitumista vaativat tilanteet voidaan helposti ja luotettavasti erottaa toisistaan ja joissa päätöksentekologiikka on yksinkertainen ja ilmaistavissa if-else -rakenteilla. Mallin eduksi voidaan laskea se, että sen avulla erotetaan toimintalogiikka päätöksentekologiikasta. [Ramirez and Cheng, 2010]

Divide and Conquer -mallissa monimutkainen uudelleenkonfiguroitumismekanismi hajotetaan useammaksi yksinkertaiseksi mekanismiksi. Malli määrittelee ensin eri mekanismien riippuvuussuhteet, luo sitten järjestyksen, joka säilyttää riippuvuudet, ja lopuksi suorittaa rinnakkain nämä mekanismi. [Ramirez and Cheng, 2010]

Architecture-based -malli perustuu arkkitehtuurimalleihin, joilla kuvataan sekä järjestelmän nykyinen arkkitehtuuri että mahdolliset muut arkkitehtuurit. Suunnittelumallissa tutkitaan jatkuvasti järjestelmän nykyistä tilaa siltä varalta, että järjestelmän ominaisuudet ovat muuttuneet. Mikäli näin on käynyt, käydään läpi muut mahdolliset arkkitehtuurimallit ja valitaan niistä sopivin, joka sitten otetaan käyttöön. [Ramirez and Cheng, 2010]

Tradeoff-based -mallissa arvioidaan mahdollisia uusia konfiguraatioita, jotka pisteytetään sen mukaan, miten järjestelmän tavoitteet toteutuvat niissä. Vaihtoehdoista valitaan sitten parhaan tuloksen saanut konfiguraatio. Suunnit-

telumallin etu on, että vaihtoehdoista valitaan se, jossa järjestelmän laatuominaisuudet ovat tasapainossa. Huonona puolena pidetään sitä, että jokaiseen laatuominaisuuteen täytyy liittää hyötyfunktio. Hyötyfunktiot ovat usein sovellusriippuvaisia, joten uudelleenkäyttö on vaikeaa. [Ramirez and Cheng, 2010]

Component Insertion -mallissa alustetaan uusi komponentti joko oletustilaan tai johonkin aikaisemmin tallennettuun tilaan. Ennen uuden komponentin lisäystä ne komponentit, jotka ovat vuorovaikutuksessa uuden komponentin kanssa, asetetaan passiiviseen tilaan. Kun uusi komponentti on lisätty järjestelmään ja yhdistetty muihin komponentteihin, passiiviset komponentit muutetaan taas aktiivisiksi. Suunnittelumallin avulla voidaan estää epäjohdonmukaisten transaktioiden suorittaminen, mutta useiden komponenttien yhtäaikainen passivoiminen voi väliaikaisesti heikentää järjestelmän suorituskykyä. [Ramirez and Cheng, 2010]

Component Removal -malli toimii samankaltaisesti kuin komponentin lisäyskin. Ennen poistoa kaikki transaktiot, joihin poistettava komponentti osallistuu, suoritetaan loppuun, jotta järjestelmän toiminta ei häiriinny, ja sekä poistettava komponentti että ne komponentit, jotka ovat yhteydessä siihen, asetetaan passiiviseen tilaan, jolloin poistettava komponentti eristetään muusta järjestelmästä eikä se voi osallistua transaktioihin. Samoin kuin Component Insertion -mallissa, myös Component Removal -mallissa komponenttien passivoiminen voi heikentää järjestelmän prosessointikykyä. [Ramirez and Cheng, 2010]

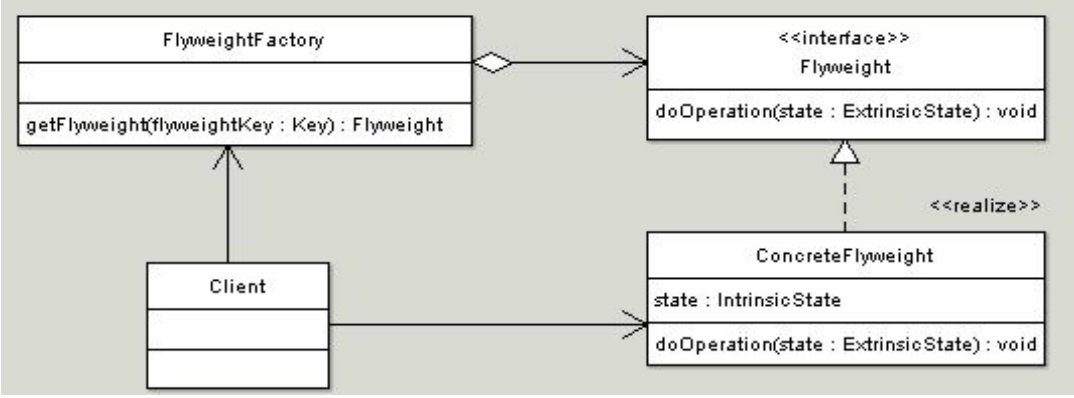
Server Reconfiguration -mallin tarkoituksena on konfiguroida palvelin uudestaan ajonaikana ilman, että asiakkaiden lähettämiä kutsuja hukataan. Malli hallitsee niiden komponenttien operatiivista tilaa, jotka muodostavat asiakas-palvelinarkkitehtuurin. Se puskuroi asiakkailta tulevat kutsut, ja ne suoritetaan kun uudelleenkonfiguraatio on valmis. Mallin avulla palvelin on johdonmukaisessa tilassa sekä ennen konfiguraatiota, sen aikana että sen jälkeen. [Ramirez and Cheng, 2010]

Decentralized Reconfiguration -mallia voidaan käyttää silloin, kun järjestelmässä ei ole keskitettyä ohjaavaa komponenttia. Tällöin jokainen komponentti osallistuu uudelleenkonfiguroitumiseen ja vastaa itsensä alustamisesta, asentamisesta sekä itsensä ja muiden komponenttien välisten yhteyksien katkaisemisesta. Mallin avulla saadaan muodostettua yhtenäinen konfiguroitumisprotokolla ja säilytetään järjestelmän johdonmukaisuus, mutta samalla uuden konfiguraation oikeellisuuden varmistaminen vaikeutuu, sillä jokainen komponentti on vastuussa itsestään. [Ramirez and Cheng, 2010]

Riva ja muut [2006] esittelevät malleja, jotka tukevat järjestelmien kontekstitietoisuutta. He ovat tutkineet erilaisia ympäristöään tarkkailevia järjestelmiä ja

etsineet niissä käytettyjä suunnittelumalleja. Tunnistetut mallit on jaettu kahteen ryhmään: niihin, jotka on esitelty Gamman ja muiden teoksessa [1995], sekä uusiin malleihin. Ennestään tunnettuja malleja ovat Flyweight, Hybrid Mediator-Observer ja Strategy. Uusia malleja ovat Flexible Context Processing ja Enactor.

Flyweight-mallia käytetään sekä havainnointiin että komponenttien säätelyyn. Se on hyödyllinen silloin, kun täytyy hallita suurta määrää sensoreita ja toimeenpanijoita. Flyweight on eri komponenttien yhteinen jäsen, jota voidaan käyttää samanaikaisesti useassa eri tilanteessa. Sen hyöty johtuu sisäisen ja ulkoisen tilan erottamisesta. Sisäinen tila koostuu tiedoista, jotka ovat riippumattomia sovelluslogiikasta, mikä mahdollistaa sisäisen tilan jakamisen. Ulkoinen tila on riippuvainen sovelluslogiikasta ja vaihtelee sen mukaan, minkä vuoksi ulkoista tilaa ei voi jakaa komponenttien kesken. Asiakasolioiden tehtävä on tarvittaessa antaa ulkoinen tila flyweight-oliolle. Kuvassa 4 on Flyweight-mallin luokkakaavio. [Riva et al., 2006]

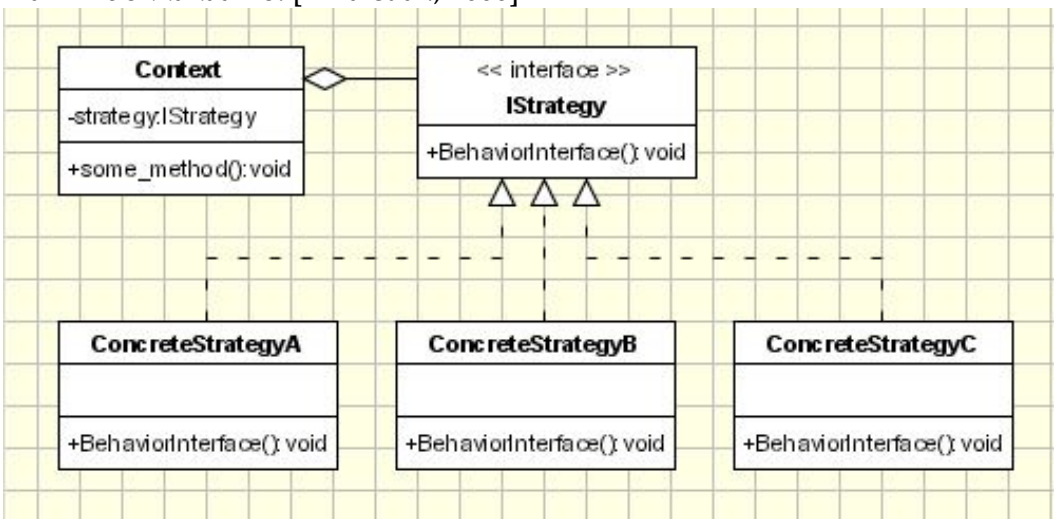


Kuva 4. Flyweight-mallin luokka kaavio [OODesign, 2012]

Hybrid Mediator-Observer -mallissa yhdistyy kaksi Gamman ja muiden [1995] esittelemää mallia. Observer-mallia käytetään järjestelmän kontekstissa tapahtuvien muutosten havaitsemiseen. Kontekstiparametrien ja Observer-komponenttien välillä on yksi moneen -suhde siten, että kun parametri muuttuu, siitä ilmoitetaan Observer-komponenteille. Parametrien ja tarkkailijoiden välisistä riippuvuuksista voi tulla erittäin monimutkaisia, kun parametrien tyyppi ja määrä muuttuu tai kun tarkkailijoiden määrä kasvaa. Riippuvuuksien vähentämiseksi käytetään Mediator-mallia, jonka avulla erotetaan toisistaan tarkkailijat ja parametrit. Yhdessä Observer- ja Mediator-mallit muodostavat Hybrid Mediator-Observer -mallin. [Riva et al., 2006]

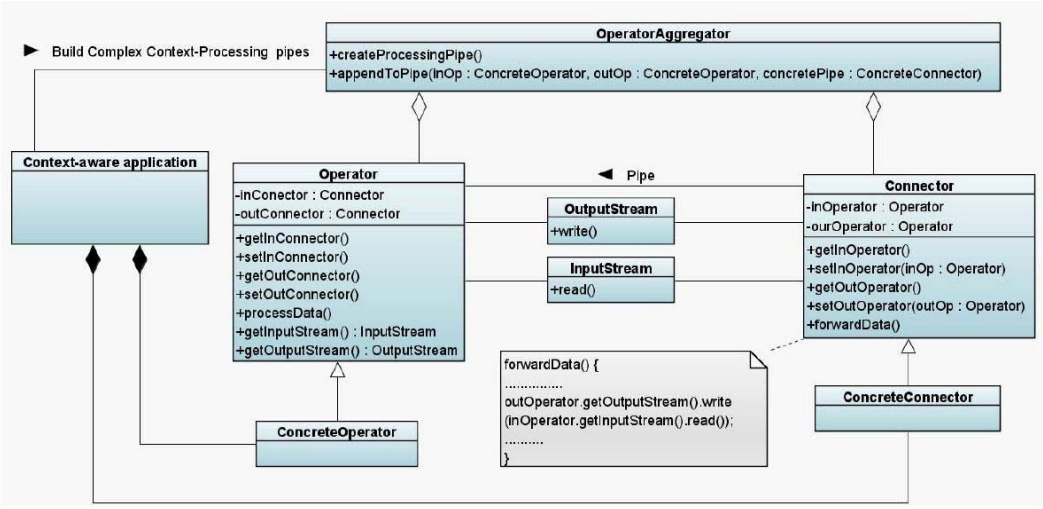
Strategy-mallia voidaan käyttää, kun järjestelmän täytyy käyttäytyä eri tavoin eri tekijöiden mukaan. Tällöin järjestelmä sisältää säännöstön, joka määrää, miten mikäkin tekijä vaikuttaa järjestelmän käyttäytymiseen. Järjestelmä

voi esimerkiksi räätälöidä käyttäytymistään käyttäjien, ympäristön tai laitteiden mukaan. Strategy-mallin avulla mahdollistetaan kontekstiriippuvaisen käyttäytymisen muunneltavuus. Järjestelmään voidaan lisätä uusia käyttäytymisstrategioita tai vanhoja voidaan muuttaa. Kuvassa 5 on esitetty Strategy-mallin luokkakaavio. [Riva et al., 2006]



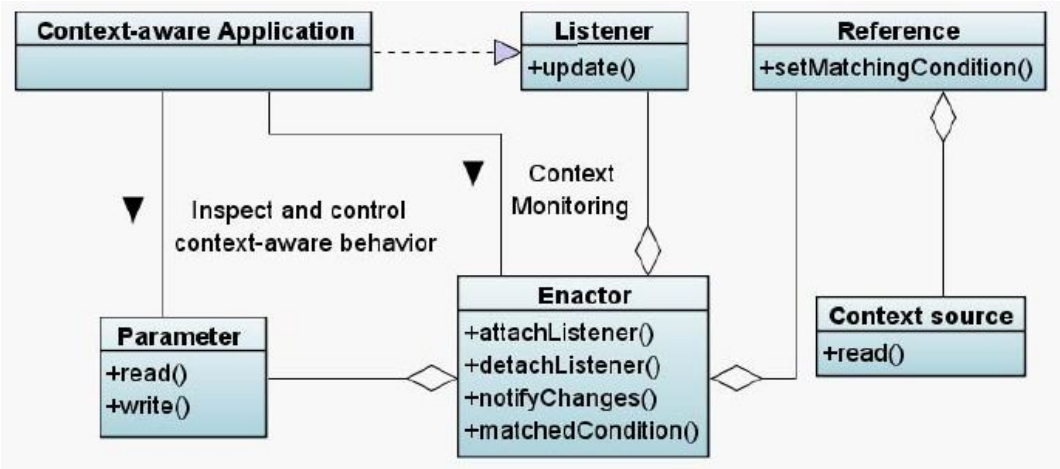
Kuva 5. Strategy-mallin luokkakaavio [OODesign, 2012]

Flexible Context Processing -mallin tarkoituksena on erottaa kontekstin prosessointi varsinaisesta sovelluslogiikasta ja tarjota siten elementtejä, joita voidaan käyttää uudestaan. Mallissa Operator-oliot lukevat tietoa syötevirrasta, prosessoivat sen ja tulostavat sen tulostevirtaan. Connector-oliot muuntavat yhden Operator-olion tulostevirran toisen Operator-olion syötevirraksi. Sovellus antaa Operator- ja Connector-komponentit OperatorAggregator-komponentille, joka järjestää ne putkeksi, jossa tieto prosessoidaan. Mallin luokkakaavio on kuvassa 6. [Riva et al., 2006]



Kuva 6. Flexible Content Processing -mallin luokkakaavio [Riva et al., 2006]

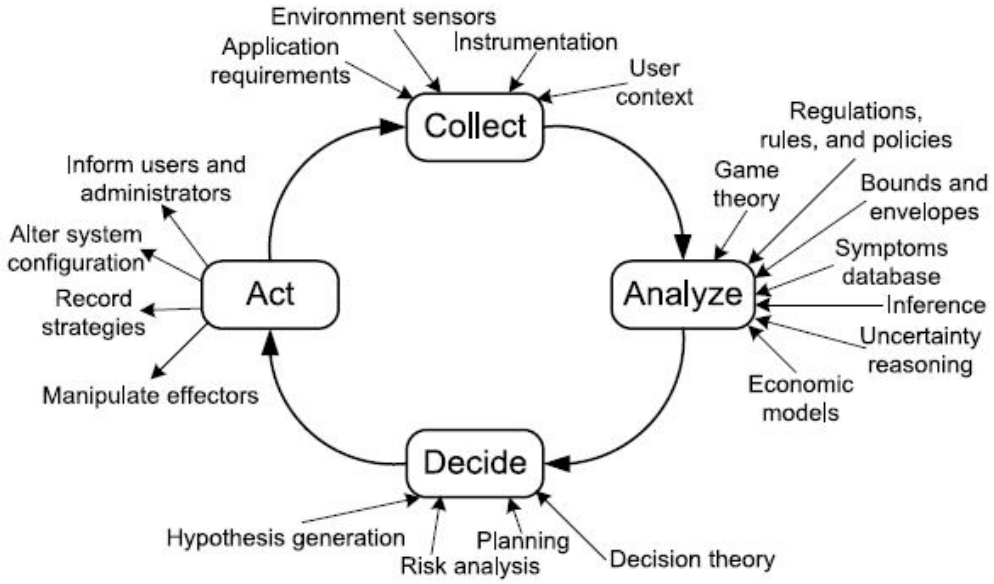
Enactor-mallissa ohjaus, tarkkailu ja toimeenpano erotetaan toisistaan. Enactor-komponentilla on kolme jäsentä: Reference-, Parameter- ja Listener-komponentit. Enactor-komponentti saa tietoa kontekstista Reference-komponentin kautta. Se prosessoi tiedon ja poimii siitä olennaisen tiedon parametreiksi. Listener-komponentti saa tiedon kaikista Enactor-komponentin tapahtumista, kuten Reference-komponentilta saadusta syötteestä tai toiminnoista, jotka Enactor käynnistää. Parameter-komponentin tehtävänä on ohjaus. Kuvassa 7 on Enactor-mallin luokkakaavio.



Kuva 7. Enactor-mallin luokkakaavio [Riva et al., 2006]

5. Ohjaussilmukat

Itseohjautuvien järjestelmien perusominaisuus itseohjautuvuus toteutetaan useimmiten jonkinlaisen ohjaussilmukan avulla. Yksi tapa toteuttaa tällainen ohjaussilmukka on MAPE-silmukka. MAPE-silmukassa on neljä komponenttia, joilla jokaisella on oma tehtävänsä. Komponenttien tehtävät ovat tarkkaileminen (monitoring), analysointi (analyzing), suunnittelu (planning) ja toimeenpano (execution). Tarkkailijakomponentti seuraa järjestelmän suoritusta ja järjestelmässä sekä sen ympäristössä tapahtuvia muutoksia. Analysointikomponentin tehtävä on analysoida, onko muutokseen reagoitava jollakin tavalla. Mikäli analyysin tuloksena järjestelmän täytyy mukautua, suunnittelijakomponentti tekee päätöksen siitä, miten järjestelmä muuttaa itseään. Toimeenpanokomponentin tehtävänä on laittaa muutos alulle. Kuvassa 8 esitetään ohjaussilmukan komponenttien toiminnot.



Kuva 8. Ohjaussilmukan toiminnot [Cheng et al., 2009]

Weyns ja muut [2012] esittelevät malleja, joilla voidaan toteuttaa itseohjautuvien järjestelmien ohjaussilmukoita. He käsittelevät sellaisia malleja, joissa ohjaustoiminnallisuus ei ole keskitettyä eli niissä on useita ohjaussilmukoita. Näitä malleja he kutsuvat nimillä Coordinated Control, Information Sharing, Master/Slave, Regional Planning ja Hierarchical Control. Näistä kahdessa ensimmäisessä käytetään rinnakkaisia ohjaussilmukoita, kun taas loput kolme perustuvat siihen, että korkeamman tason ohjaussilmukat kontrolloivat alemman tason ohjaussilmukoita.

Coordinated Control -mallia voidaan käyttää silloin, kun mukautumiseen tarvittava tieto on hajanaisesti järjestelmässä, järjestelmä on mittakaavaltaan niin suuri, että tiedon käsittely keskitetysti ei kannata, tai kun järjestelmä ulottuu monelle sellaiselle alueelle, joilla ei ole luotettavaa tahoa kontrolloimassa mukautumista. Tällaisissa tilanteissa jokaisella rinnakkaisella järjestelmän osalla on oma ohjaussilmukkansa, joiden MAPE-komponentit kommunikoivat rinnakkaisten silmukoiden vastaavien komponenttien kanssa. Coordinated Control -mallin etuna on sen skaalautuvuus. Lisäksi se voi tehdä järjestelmästä vakaamman, sillä yhden osakomponentin kaatuminen ei kaada koko järjestelmää. Ongelmaksi voi sen sijaan muodostua se, että ei voida taata kaikkien ohjaussilmukoiden toteuttavan oman ohjattavan järjestelmänsä mukautumista yhdenmukaisesti. Lisäksi rinnakkaisten ohjaussilmukoiden yhdessä tekemät mukautumispäätökset saattavat johtaa siihen, että järjestelmä kokonaisuutena ei enää toimi optimaalisella tavalla. [Weyns et al., 2012]

Information Sharing -malli on hyödyllinen silloin, kun järjestelmän komponentit ovat löyhästi yhteydessä toisiinsa ja kun järjestelmän osat voivat mukautua paikallisesti, mutta tarvitsevat tietoa muiden osien tilasta, koska paikalliset muutokset voivat vaikuttaa muihinkin osiin. Tässä mallissa ainoastaan rinnakkaisten ohjaussilmukoiden tarkkailijakomponentit kommunikoivat keskenään. Muut komponentit kommunikoivat vain saman silmukan muiden komponenttien kanssa. Information Sharing -mallin etuja ovat sen skaalautuvuus ja ajantasaisuus. Koska analyysi-, suunnittelu- ja toimeenpanokomponentit eivät kommunikoi ulkopuolisten komponenttien kanssa, järjestelmä kykenee tekemään päätökset ja käynnistämään mukautumismekanismin ajantasaisesti. Huonona puolena voidaan pitää sitä, että paikalliset mukautumiset voivat vaikuttaa negatiivisesti globaalissa mittakaavassa. Paikallisesti tehdyt päätökset voivat myös olla ristiriidassa toistensa kanssa, mikä johtaa pysyviin muutoksiin järjestelmässä. Tämä voi vaikuttaa resurssien käyttöön sekä järjestelmän saavutettavuuteen ja vakauteen. [Weyns et al., 2012]

Master/Slave -mallissa on yksi suunnittelijakomponentti ja yksi analyysikomponentti. Tarkkailija- ja toimeenpanokomponentteja voi olla useampia. Jokainen tarkkailijakomponentti kommunikoi analyysikomponentin kanssa ja jokainen toimeenpanokomponentti suunnittelijakomponentin kanssa. Analyysi- ja suunnittelijakomponentit kommunikoivat keskenään. Tätä mallia voidaan käyttää, kun tiedon kerääminen ja mukautuminen täytyy tapahtua paikallisesti, mutta paikalliset muutokset vaikuttavat myös muiden osajärjestelmien toimintaan. Mallin etuna on se, että analysoinnin ja suunnittelun keskittäminen avulla globaalien tavoitteiden saavuttaminen on tehokkaampaa. Keskittämisestä voi kuitenkin olla myös haittaa, sillä varsinkin suuremman mittakaavan hajautetussa järjestelmässä tiedon prosessointi voi olla hidasta. [Weyns et al., 2012]

Regional Planning -mallia voidaan käyttää silloin, kun järjestelmässä on useita löyhästi toisistaan riippuvaisia osia, jotka vaativat paikallista mukautumista mutta myös osien välistä mukautumista. Jokaisella osalla on yksi suunnittelijakomponentti, joka suunnittelee paikallisen mukautumisen ja joka kommunikoi muiden osien suunnittelijakomponenttien kanssa suunnitellakseen osien välisen mukautumisen. Yhdessä osassa on yksi tai useampia tarkkailija-, analyysi- ja toimeenpanokomponentteja. Tarkkailijat keräävät tietoa osan alijärjestelmistä, analyysikomponentit analysoivat tiedon ja raportoivat sen suunnittelijakomponentille, joka yksin tai yhdessä muiden suunnittelijoiden kanssa päättää, miten järjestelmän täytyy mukautua, ja välittää tiedon tarvittaville toimeenpanokomponenteille. Regional Planning -mallin etuna on se, että sen avulla yhden osan sisällä eri ohjaussilmukat voivat keskittyä eri asioihin, mutta suunnittelu tapahtuu kuitenkin ylemmällä tasolla. Lisäksi paikallinen tarkkailu

ja analyysi voivat vähentää vuorovaikutusta suunnittelijakomponentin kanssa, mikä parantaa suorituskykyä. Riskinä on kuitenkin, että mukautuminen ei ole tehokasta. Lisäksi mukautumisen toimeenpano täytyy suunnitella huolellisesti, sillä malli ei tue toimeenpanokomponenttien välistä vuorovaikutusta. [Weyns et al., 2012]

Hierarchical Control -mallia voidaan käyttää silloin, kun ohjaava alijärjestelmä on monimutkainen ja sen itsensä täytyy mukautua. Siinä on useita ohjaussilmukoita, jotka hallitsevat eri resursseja tai joiden toiminta-aika on erilainen. Niiden täytyy kuitenkin kommunikoida keskenään välttääkseen ristiriitoja. Hierarchical Control -mallissa ohjaussilmukat ovat hierarkkisessa suhteessa toisiinsa. Eri kerrosten silmukat hallitsevat eri asioita eri abstraktiotasoilla. Alimpien kerrosten silmukat toimivat lyhyellä aikavälillä ja ylimmän kerroksen silmukat pitkällä aikavälillä. Alimman kerroksen silmukat hallitsevat ohjattavaa alijärjestelmää ja välikerrosten silmukat alempia kerroksia. Kaikkein ylimmällä tasolla oleva silmukka on vastuussa koko järjestelmän ohjaamisesta. Alimman tason silmukoiden tarkkailija- ja toimeenpanokomponentit kommunikoivat ohjattavan alijärjestelmän kanssa ja sitä ylempien tasojen vastaavat silmukat ovat vuorovaikutuksessa niiden alapuolella olevan kerroksen silmukoiden kanssa. Kaikkien tasojen analyysi- ja suunnittelijakomponentit ovat vuorovaikutuksessa ainoastaan oman silmukkansa muiden komponenttien kanssa. [Weyns et al., 2012]

6. Yhteenveto

Tarve järjestelmille, jotka kykenevät mukautumaan niissä itsessään tai toimintaympäristössä tapahtuvien muutosten mukaan, on kasvamassa, sillä järjestelmien ylläpito vie paljon resursseja ja aiheuttaa lisäkustannuksia. Ylläpitokustannuksia voidaan vähentää, jos järjestelmä pystyy itse optimoimaan suorituskykyään, korjaamaan virheitä tai muulla tavoin ylläpitämään toimintakykyään.

Tutkielmassa käsiteltiin itseohjautuvien järjestelmien suunnittelua ja rakennetta. Itseohjautuvien järjestelmien suunnitteluun vaikuttavat järjestelmän tavoitteet, mukautumisen taustalla olevat muutokset, mekanismit, joiden avulla järjestelmän mukautuminen toteutetaan sekä mukautumisen vaikutukset. Tutkielmassa kartoitettiin, millaisilla suunnitteluratkaisuilla järjestelmiin voidaan toteuttaa itseohjautuvia ominaisuuksia.

Lisää itseohjautuvien järjestelmien suunnittelua ja toteutusta tukevia malleja ja menetelmiä kuitenkin tarvitaan. Koska itseohjautuvuus käsittää paljon erilaisia ominaisuuksia, tulisi selvittää, millaisilla malleilla ja mekanismeilla mitäkin niistä voidaan toteuttaa. Tämä helpottaisi oleellisesti järjestelmäarkkitehtien

työtä ja nopeuttaisi järjestelmien suunnittelua vähentäen näin järjestelmien kokonaiskustannuksia.

Viiteluettelo

[Anderson et al., 2009] Jesper Anderson, Rogério de Lemos, Sam Malek and Danny Weyns, Modeling dimensions of self-adaptive software systems. In: B. H. C. Cheng et al. (eds.) *Software Engineering for Self-Adaptive Systems*. Dagstuhl Seminar, 2009, 27-47.

[Cheng et al., 2009] Betty H. C. Cheng, Rogério de Lemos, Holger Giese, Paola Inverardi and Jeff Magee, Software engineering for self-adaptive systems: A research roadmap. In: B. H. C. Cheng et al. (eds.) *Software Engineering for Self-Adaptive Systems*. Dagstuhl Seminar, 2009, 1-26.

[Gamma et al., 1995] Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides, *Design Patterns. Elements of Reusable Object-Oriented Software.* Addison-Wesley, 1995.

[Gomaa and Hussein, 2004] Hassan Gomaa and Mohamed Hussein, Software reconfiguration patterns for dynamic evolution of software architectures. In: *Proc. of the 4th Working IEEE/IFIP Conference on Software Architecture,* 79-88.

[Gorla, 2007] Alessandra Gorla, Towards design for self-healing. In: *Proc. of the Fourth International Workshop on Software Quality Assurance*, 86-89.

[Medvidovic and Mikic-Rakic, 2001] Nenad Medvidovic and Marija Mikic-Rakic, Architectural support for programming-in-the-many. Technical report USC-CSE-2001-506. Available as http://csse.usc.edu/ csse/ TECHRPTS/2001/usccse2001-506/usccse2001-506.pdf.

[Mühl et al., 2007] Gero Mühl, Matthias Werner, Michael A. Jaeger, Klaus Herrmann and Helge Parzyjegla, On the definitions of self-managing and self-organizing systems. In: *Proc. of the 2007 ITG-GI Conference of Communication in Distributed Systems (KiVS)*, 1-11.

[OODesign, 2012] Object Oriented Design, Design patterns. http://www.oodesign.com/. Checked 8.12.2012.

[Oreizy et al., 2008] Peyman Oreizy, Nenad Medvidovic and Richard N. Taylor, Runtime software adaptation: Framework, approaches, and styles. In: *Companion of the 30th International Conference on Software Engineering,* 899-910.

[Ramirez and Cheng, 2010] Andres J. Ramirez and Betty H. C. Cheng, Design patterns for developing dynamically adaptive software. In: *Proc. of the 2010 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems,* 49-58.

[Riva et al., 2006] Oriana Riva, Cristiano di Flor, Stefano Russo and Kimmo Raatikainen, Unearthing design patterns to support context-awareness. In: *Proc. of Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, 383-387.

[Salehie and Tahvildari, 2009] Mazeiar Salehie and Ladan Tahvildari, Self-adaptive software: Landscape and research challenges. *ACM Transactions on Autonomous and Adaptive Systems,* **4**, 2 (May 2009), Article No. 14.

[Serugendo et al., 2004] Giovanna Di Marzo Serugendo, Noria Foukia, Salima Hassas, Anthony Karageorgos, Soraya Kouadri Mostéfaoui, Omer F. Rana, Mihaela Ulieru, Paul Valckenaers and Chris Van Aart, Self-organization: paradigms and applications. In: G. Di Marzo Serugendo, et al. (eds.) *Engineering Self-Organizing Systems, LNCS* **2977**, Springer 2004, 1-19.

[Taylor et al., 1995] Richard N. Taylor, Nenad Medvidovic, Kenneth M. Anderson, E. James Whitehead Jr. and Jason E. Robbins, A component- and message-based architectural style for GUI software. In: *Proc. of 17th International Conference on Software Engineering*, 295-304.

[Weyns et al., 2012] Danny Weyns, Bradley Schmerl, Vincenzo Grassi, Sam Malek, Raffaela Mirandola, Christian Prehofer, Jochen Wuttke, Jesper Anderson, Holger Giese and Karl Göschka, On patterns for decentralized control in self-adaptive systems. In: R. de Lemos, et al. (eds.) *Self-Adaptive Systems, LNCS* **7475**, Springer 2012, 76-107.

Ylläpito ja lisenssit avoimissa ohjelmistoissa

Hanne Korhonen

Tiivistelmä.
Tässä tutkielmassa perehdytään tietojärjestelmien ylläpitoon ja pohditaan, onko tietojärjestelmien ylläpidolla ja avoimien ohjelmistojen lisensseillä yhteyttä, sekä mietitään, pitäisikö niillä olla yhteyttä. Johtopäätöksenä todetaan, ettei lisensseihin kannata lisätä ylläpitoon liittyviä määräyksiä tai ohjeita, koska se ei ole lisenssien tarkoitus.

Avainsanat ja -sanonnat: tietojärjestelmien ylläpito, avoimet ohjelmistot, lisenssit.
CR-luokat: K.6.3

1. Johdanto

Suurien tietojärjestelmien ylläpitäminen on erittäin hankalaa, monimutkaista sekä aikaa vievää ja ylläpito muuttuu ajan myötä vain vaikeammaksi [Godfrey and Tu, 2000]. Avoimilla ohjelmistoilla kasvu on nopeaa ja ne muuttuvat helposti monimutkaisiksi, mutta niiden ylläpitoon ei tunnuta kiinnitettävän tarpeeksi huomiota. Varsinkin ylläpidon hallinnointi on minimaalista.

Tässä tutkielmassa tutkitaan käytetyimpiä avoimien ohjelmistojen lisenssejä, joista tutkitaan erityisesti avoimien lisenssien määräyksiä ylläpidosta. Lisenssit on valittu Open Source Iniativen www-sivuilta [OSI, 1998] kategoriasta suosituimmat ja käytetyimmät lisenssit. Kirjallisuuskatsaukseni perusteella näyttää, että avoimien ohjelmistojen lisenssejä ylläpidon kannalta on tutkittu melko vähän.

Tutkin asiaa etsimällä lisenssiehdoista mainintoja ylläpitoon liittyvistä asioista. Hyvä puoli tutkimustavassani on se, että se on nopea ja selkeä tapa tutkia lisenssejä. Lopuksi pohdin, onko avoimien ohjelmistojen lisenssit hyvä tapa sisällyttää ylläpito tietojärjestelmiin ja ohjelmistotuotteisiin.

Tulokseksi nousee se, ettei lisensseissä oteta juurikaan ylläpitoa huomioon. Muutamissa lisensseissä mainitaan siitä, että omat muutokset pitää merkitä selkeästi, tai viitataan epäsuorasti ylläpitoon, mutta mitään kunnollisia ohjeita tai standardeja ylläpitoa varten ei kerrota. Olen myös tarkastelun tuloksena huo-

mannut, ettei ole edes järkevää sisällyttää ylläpito-ohjeita avoimien ohjelmistojen lisensseihin.

Luvussa 2 käydään läpi tietojärjestelmien ylläpidon tyypillisiä piirteitä sekä sitä, millaisia ohjeita muissa tutkimuksissa on annettu, jotta ylläpidosta tulisi toimivampaa. Luvussa 3 tutkitaan, millaisia avoimien ohjelmistojen lisenssit ovat, sekä huomioidaan tärkeitä piirteitä avoimien ohjelmistojen ylläpidon kannalta. Luvussa 3 tutkitaan myös avoimien ohjelmistojen lisenssien ja ylläpidon yhdistämistä. Viimeisessä luvussa eli yhteenvedossa käydään läpi tulokset ja pohditaan asiaa tulevaisuuden kannalta.

2. Tietojärjestelmien ylläpito

Tietojärjestelmien ylläpito kestää tietojärjestelmien koko elinkaaren ajan [Bennet and Rajlich, 2000]. Monet käyttävät mieluummin ylläpidosta puhuessaan ilmausta *tietojärjestelmien evoluutio* (software evolution) [Bennet and Rajlich, 2000], vaikka monesti puhutaan myös tietojärjestelmän kehittämisestä. Nämä molemmat termit kuitenkin useimmiten tarkoittavat samaa asiaa. Termien erilainen käyttö kuitenkin vaikeuttaa asian ymmärtämistä sekä eri termin käyttö vaikuttaa asenteeseen ja suhtautumiseen.

”Ylläpito” tarkoittaa sitä, että tuote pidetään toimintakunnossa sen jälkeen, kun se on alun perin julkaistu. Sen tulee toimia oikein jo heti alusta asti, niin kuin alun perin on suunniteltu [Parikh, 1986]. Kuitenkin voi olla, ettei ohjelmistotuote toimi oikein alun alkaenkaan; ylläpito ei silloin ole ylläpitoa vaan virheiden korjaamista ja testaamista eli kehitystyötä. Tätä työtä ei pitäisi Parikhin [1986] mukaan kutsua ylläpidoksi. Mutta nykyään näin tehdään.

Yleensä tietojärjestelmien ylläpito määritellään tarkoittavan ohjelmistotuotteen muuttamista sen jälkeen, kun se on toimitettu asiakkaalle. Ohjelmistotuotteen muuttamista on virheiden korjaaminen, suorituskyvyn tai muiden ominaisuuksien parantaminen sekä tuotteen muokkaaminen muutettuun ympäristöön. [Mamone, 1994]

Tietojärjestelmien evoluutiolle ei ole olemassa virallista määritelmää, mutta sitä käytetään usein synonyyminä tietojärjestelmien ylläpidolle. Chapin ja muut [2001] määrittelevät tietojärjestelmien evoluution näin:

> ”Joskus sitä käytetään synonyyminä tietojärjestelmien ylläpidolle ja joskus kuvailemaan tapahtumia ja muutoksia, jotka tapahtuvat alkuperäisen luonnin ja eläkkeelle jäämisen välissä.”

Bennet ja Rajlich [2000] ovat sitä mieltä, että ylläpidettävyydelle tarvitaan parempi määritelmä, koska tarvitaan hyvä teoreettinen tausta ja ymmärrys yllä-

pidolle. Heidän mukaansa tietojärjestelmien ylläpidon tavoite on muuttaa olemassa olevaa ohjelmistotuotetta samalla kun säästetään sen yhtenäisyys. Tähän tavoitteeseen kuitenkin harvemmin päästään. Olio-ohjelmoinnin uskottiin olevan ratkaisu tietojärjestelmien ylläpitoon, mutta siitä seurasikin uusia erilaisia ylläpidon ongelmia. [Bennet and Rajlich, 2000]

Tässä tutkielmassa ylläpidon määritelmänä käytetään Mamonen [1994] määritelmää, koska sama määritelmä on laajasti käytössä muissakin tutkimuksissa [Bhatt et al., 2004; Bennet and Rajlich, 2000].

Muutoksia on hankala ennakoida, koska jotkin muutokset järjestelmään ovat sellaisia, joita alkuperäiset suunnittelijat eivät voi olla edes tietoisia [Bennet and Rajlich, 2000], koska teknologia muuttuu ajan myötä. Siksi tietojärjestelmien ylläpito ei koskaan katoa. On hyvin epätodennäköistä, että keksitään sellainen järjestelmä, joka osaa itse korjata kaikki viat ja ongelmat.

2.1. Dokumentaatio

Dokumentti tarkoittaa mitä tahansa kirjoitettua dokumenttia tietojärjestelmästä, mukaan lukien lähdekoodi, viestintä, raportit sekä muodollinen dokumentaatio [Das et al., 2007].

Usein tutkimuksissa kerrotaan, kuinka yksi suurimpia tietojärjestelmien ongelmia on juuri dokumentaatio ja sen puute [Mamone, 1994; Das et al., 2007; Bennet and Rajlich, 2000; Yip et al., 1994]. Dokumentaatio on usein puutteellista, eikä se ole ajan tasalla tai sitä ei ole ollenkaan olemassa [Mamone, 1994]. Muita dokumentaation ongelmia ovat tarkkuus, ymmärrettävyys ja käypäisyys [Das et al., 2007]. Jos dokumentti on liian vaikeaselkoinen tai pitkä, sitä on myös vaikeampi päivittää oikeaan muotoon.

Das ja muiden [2007] mukaan ylläpitäjät käyttävät tietolähteinä muita ihmisiä, sekä kirjoitettuja ja elektronisia lähteitä. Informaation kerääminen on aikaa vievin työtehtävä tietojärjestelmän ylläpidossa. Lähdekoodi ei ollut aina ohjelmoijien ensimmäinen valinta etsiä tietoa, yleensä se on viimeinen keino. Kuitenkin se on luotetuin, päivitetyin ja saatavilla, muttei kuitenkaan helpoin tulkita. Singer [1998] väittää, että lähdekoodi on ylläpitäjille käytännössä Raamattu. Ylläpitäjät ovat ylläpitämiensä ohjelmistojen erikoisasiantuntijoita, joille lähdekoodi on päätiedonlähde. Myös dokumentaatiota käytetään, mutta siihen ei luoteta. [Singer, 1998] Dokumentin sijoittamisen paikka on tärkeä. Ratkaisu saattaa olla dokumentissa, josta ylläpitäjä ei tiedä mitään tai jota hän ei löydä. [Das et al., 2007; Lutters and Seaman, 2007; Singer 1998]

Jos dokumenttia ei löydy tai ole olemassa, käytetään apuna henkilöitä. Muiden ihmisten neuvoja pidetään luotettavana tiedonlähteenä, kun etsitään

ihmisiä, jotka tietävät ylläpidettävästä ohjelmasta. Usein sellainen epävirallisena tai epämuodollisena pidetty dokumentti onkin pelastava tekijä, kun kohdataan vaikea ylläpitotehtävä. [Lutters and Seaman, 2007]

Koska tietojärjestelmien koodin uudelleen käyttäminen kaksinkertaistuu, kun käytetään olio-ohjelmointia [Mancl and Havanas, 1990], on tärkeää, että dokumentointi on olemassa ja ajan tasalla. Näin järjestelmään on helpompaa lisätä uusia toimintoja vähemmillä rajapintamuutoksilla. Bennet ja Rajlich [2000] paljastavat, että ylläpito suoritetaan todellisuudessa käyttämällä lähdekoodia, mikä ei ole ihanteellista, jotta päästäisiin mahdollisimman tehokkaaseen ylläpitoon.

Koposen ja Hotin [2005] mukaan ylläpitoa pitäisi suunnitella ja suunnitelmia pitäisi kehittää ja päivittää tarpeiden mukaan. Suunnitelmat voivat olla epävirallisia, mutta on tärkeää, että jossain näkyy, miten ylläpitäminen edistyy. Säännöllinen päivittäminen dokumenteille ja järjestelmälle on järkevää. On mahdollista, ettei ylläpitäjä tunne itse edistyvänsä ylläpidossa, joten on tärkeää, että on olemassa jonkin näköinen raportti siitä mitä on tehty [Koponen and Hotti, 2005]. Jos pitää suunnitelmia jossain kirjoitettuna, saa niistä helpommin muotoiltua myöhemmin dokumentin, josta on hyötyä jos ylläpitäjä vaihtuu.

2.2. Asenne

Jo sanana ylläpito ei kuulosta kovin miellyttävältä tai kiinnostavalta. Asenneongelma on yksi syy, miksei tietojärjestelmien ylläpito toimi. Asenteen näkee jo jostain kirjoituksista. Sharon [1996] kirjoittaa näin:

> ”Ohjelmoijien tapauksessa vähemmän taitavat on alennettu ylläpitotehtäviin sillä välin kuin lahjakkaille on annettu vaativampi uuden systeemin kehittämistyö.”

Lainauksesta näkee hyvin, kuinka ylläpitoa pidetään vähempiarvoisena kuin kehitystyötä, vaikka usein tilanne on se, että työ voi olla hyvinkin samanlaista. Ylläpitäminen voi myös olla paljon hankalampaa kuin kehitystyö.

Tinnirello [1983] on sitä mieltä, että ylläpidon ongelmat voi jakaa kolmeen kategoriaan: ylläpidon hallintaan, ylläpidon ohjelmointiin sekä asenteeseen. Hän väittää, ettei minkäänlaista kehitystä tapahdu ylläpidossa, ellei jokaista näistä ongelmista paranneta. Hänen mielestään tosiasia on se, ettei ylläpito vain katoa.

Asiakkaan ja päälliköiden asenne ylläpitoa kohtaan vaikuttaa suuresti ylläpitäjien motivaatioon ja tuottavuuteen, joka vaikuttaa suoraan työn aiheuttamaan vaivannäköön ja laatuun. Ohjelmoijan asenne on yksi sivuutetuimmista ja

tärkeimmistä ominaisuuksista, jotka vaikuttavat ylläpidon laatuun ja tuottavuuteen. [Bhatt et al., 2004]

Andrews ja Lutifiyya [2000] pitivät kurssin ja tutkivat, kuinka opiskelijoilta onnistuu tietojärjestelmien ylläpito. He huomasivat, ettei yliopistoissa pidetä ylläpitoon liittyviä kursseja ollenkaan. He uskovat syiksi näitä: asenne, tutkimuksen puute sekä se, että kurssi vaatisi myös jonkun oikean projektin, jota opiskelijat voisivat ylläpitää. Taylor ja muut [2001] huomasivat myös, että oikeanlaista ja tarpeellista koulutusta ylläpitäjille ei tarjota juuri lainkaan monissa IT-alan yrityksissä. Wang ja muiden [2001] mukaan tärkein tekijä ylläpidossa on yksittäisen ohjelmoijan kyvyt. Hyvä koulutus ja harjoitus ohjelmistotuotantoon on avaintekijä onnistuneeseen toistuvaan ylläpitämiseen [Wang et al., 2001].

2.3. Hyvät ylläpitokäytännöt

Kaikki tietävät, että ylläpito on hankalaa ja aikaa vievää, koska ylläpitäjät eivät ole mukana kehittämässä alkuperäistä tuotetta. He eivät siis ymmärrä sitä kunnolla ja tuloksena ylläpito on ”huonompaa” kuin alkuperäinen koodi. Wang ja muut [2001] kuitenkin huomasivat, ettei ylläpitäjän ja alkuperäisen koodin eroa ole kun tutkitaan huonoja liitoksia. He uskovat, että syynä huononevaan koodin on se, että yritykset asettavat heikommat ohjelmoijat ylläpitotehtäviin, mikä selittää miksi toistuvasta ylläpidosta seuraa huonolaatuista koodia. [Wang et al., 2001]

Ylläpitoa voi parantaa jo usein alkuperäisessä kehitysvaiheessa, jos suunnittelussa on otettu huomioon ohjelmiston tulevaisuus. Ohjelmistokehittäjät voivat parantaa ohjelmistotuotteen ylläpidettävyyttä vähentämällä esimerkiksi tarpeettomia muuttujia, ulkoisia tietoelementtejä, ulkoisiin funktioihin viittaamista jo hyvin varhaisessa kehitysvaiheessa [Kozlov et al., 2008].

Taulukossa 1 on listattu kolmen eri tutkimuksen suurimpia ongelmia tietojärjestelmien ylläpidosta. Eroavaisuudet listauksissa johtuvat varmasti aikaerosta sekä siitä, että oli tutkittu erilaisia ohjelmistoja. Samoja ongelmia kuitenkin kaikista kolmesta listoista löytyy, ne on vain kirjoitettu hieman eri muotoon. Mielenkiintoista uusimmassa listauksessa on se, että mukana ei ole huonoa dokumentaatiota. Syynä voi olla se, että dokumentaatio on parantunut vuosien saatossa, jota hieman epäilen, tai sitten se, että on ajateltu ettei dokumentaatio ole tärkeää. Tärkeämpää on saada järjestelmä ja lähdekoodi sellaiseen muotoon, ettei dokumentaatiota edes tarvittaisi.

Kajko-Mattsson [2004]	Lientz [1983]	Yip ja muut [1994]
Sovellusten monimutkaisuus	Dokumentaation laatu	Huono dokumentaatio
Asiakkaan tietämättömyys	Käyttäjien vaatimukset parannuksille ja laajennuksille	Vaihtuvat käyttäjävaatimukset
Tukiorganisaatioiden koko ja monimutkaisuus	Ylläpitäjän työajan kilpailevat vaatimukset	Vaihtuva henkilöstö
Alustojen monimutkaisuus	Vaikeus toteuttaa aikataulutus sitoumukset	
Organisaatiouudistukset	Liikevaihto käyttäjien organisaatioissa.	
Henkilöstöriippuvuus		
Asiakkaan väärät odotukset		

Taulukko 1: Muutama listaus tietojärjestelmien ylläpidon ongelmista

Kajko-Mattsson [2004] listasi ongelmia käyttöliittymien ylläpitoa tukevien organisaatioiden ylläpidossa. Monimutkaisuuden poistaminen on hankalaa, mutta hyvällä suunnittelulla sen pitäisi vähentyä. Asiakkaiden vähäinen tieto ylläpidosta ja ohjelmistotuotteen kehittämisestä on ymmärrettävää. Usein asiakkaita ei edes kiinnosta tietää, he haluavat vain tietynlaisen järjestelmän mahdollisimman nopeasti ja mahdollisimman halvalla. Ainoa mahdollisuus korjata tätä ongelmaa on neuvoa asiakasta ja kertoa asiat niin kuin ne todellisuudessa ovat, eikä luvata mitään sellaista, mitä ei oikeasti pysty tai ehdi toteuttaa. Henkilöstöriippuvuus on myös sellainen ongelma, jota on vaikea korjata. Paras mahdollinen tapaus olisi se, että samat henkilöt, jotka tekivät järjestelmän, jäisivät myös sitä ylläpitämään, mutta tämähän on käytännössä mahdotonta. Hyviä ohjelmoijia ei kannata sitoa yhteen järjestelmään kymmeniksi vuosiksi.

Lientz [1983] listaamat ongelmat vaikuttavat sellaisilta asioilta, jotka luulisi olevan helppo korjata. Dokumentaatio paranee vain sitä tekemällä, kun sitä korjaa tarpeeksi monesti, on luultavaa, että dokumenttien laatu paranee. Ongelmana tässä varmaankin on se, ettei dokumenttien kirjoittamiseen ole tarpeeksi tai ollenkaan aikaa ja siksi niiden taso on huono. Käyttäjien toivomat vaatimuk-

set ja parannukset pitää osata priorisoida ja kategorisoida niin, että kaikista tärkeimmät toteutetaan. Kaikkia toivomuksia ei edes pidä lähteä toteuttamaan, vaan pysyä sellaisissa, jotka ovat tietojärjestelmän tarkoituksen mukaisia.

Desharnais ja April [2010] ovat sitä mieltä, että jos käyttää erinomaisia ohjelmointikäytäntöjä, niin sekä ohjelmiston kypsyyden ja valmiuksien pitäisi edistyä. Mutta kaikkea on aina helpompi tehdä teoriassa kuin käytännössä. Usein työryhmillä on liian kiireiset aikataulut suunnitella ja toteuttaa suunnitelma erinomaisilla ohjelmointikäytännöillä. Ohjelmistotuote voi olla huono, mutta se ei välttämättä tarkoita sitä, että ylläpitoryhmä on tehnyt huonoa työtä [Desharnais and April, 2010]. Se voi tarkoittaa myös sitä, että alkuperäisessä kehityksessä ei ole tehty kunnolla jotain tärkeää vaihetta, esimerkiksi arkkitehtuurin suunnittelua tai vaatimusten määrittelyä.

Eierman ja Dishaw [2007] huomattavat, ettei ole paljon empiiristä aineistoa siitä, että olio-ohjelmointiin perustuva ohjelmointikieli parantaisi ylläpidettävyyttä verrattaessa kolmannen sukupolven ohjelmointikieliin. Ohjelmointikielellä vaikuttaa ylläpitoprosessiin, sillä oli vaikutus vaivannäköön erilaisissa kognitiivisissä tehtävissä, joita vaaditaan ylläpidossa [Eierman and Dishaw, 2007].

Schneider [1983] ehdottaa ylläpidon helpottamiseksi sitä, että ylläpitotehtävät jaetaan neljään eri roolin. Tämä ei kuitenkaan ole innostanut ylläpitäjiä, koska harvemmin ylläpitoryhmässä on edes neljää henkilöä. Teoriassa idea on hyvä, mutta yrityksillä ei ole resursseja palkata niin montaa henkilöä toteuttamaan ylläpitoa.

Scheinewind [1987] neuvoo ylläpitäjiä opettelemaan lukemaan ohjelmaa ja lähdekoodia, pitämään päiväkirjaa virheistä ja ongelmista sekä ottamaan aina huomioon tietojärjestelmän seuraavan vaiheen sen elinkaaresta, koska tietojärjestelmän tulee kehittyä ja muuttua.

On hyödyllistä ennakoida ja tehdä ylläpito helpommaksi muuttamalla vaikeasti ymmärrettävä koodi moderniksi ohjelmointikieleksi ja päivittää tietokannat sekä yksinkertaistaa koodin rakenne [Bennet and Rajlich, 2000].

On mielenkiintoista, että neljän eri ylläpitotyypin välillä ohjelman sisäisten huonojen riippuvuuksien määrä vaihtelee huomattavasti. Wang ja muut [2001] käyttivät tyyppeinä korjaavaa, täydellistävää, mukautuvaa sekä muita. Täydellistävä ylläpitäminen aiheutti eniten huonoja riippuvuuksia koodissa. Korjaavassa ylläpitämisessä korjataan jo olemassa olevia toimintoja. Täydellistävässä lisätään uusia ominaisuuksia, ja he uskovat, että ohjelmoijat tekivät uudet ominaisuudet mahdollisimman nopeasti, ottamatta huomioon hyviä ohjelmointikäytäntöjä. [Wang et al., 2001]

Bennet ja Rajlich [2000] kertovat myös, kuinka kykenemättömyys muuttaa järjestelmää nopeasti ja luotettavasti tarkoittaa, että liiketoiminta mahdollisuudet ovat mennyttä. Toimiva tietojärjestelmien ylläpito on keskeinen, jotta saavutetaan maailmanlaajuinen kilpailukyky ohjelmistotuotannossa [O'Neill, 1997].

3. Avoimet ohjelmistot

Tärkein vaatimus avoimissa ohjelmistoissa on se, että lähdekoodi tulee olla vapaasti saatavilla kaikille, jotka haluavat tutkia ja muuttaa sitä omiin tarkoituksiin sopivaksi [Godfrey and Tu, 2000]. OSI (Open Source Iniative) määrittelee avoimet ohjelmistot seuraavilla kriteereillä:

1) ohjelmaa pitää voida vapaasti levittää
2) ohjelman täytyy sisältää lähdekoodia ja ohjelman levitys täytyy olla sallittu sekä lähdekoodina, että käännetyssä muodossa
3) lisenssin on sallittava muutokset ja johdettujen ohjelmistojen sallitaan saman lisenssin alle kuin alkuperäinen työ
4) lähdekoodin tulee olla yhteen kuluva
5) ei saa syrjiä ketään henkilöä tai henkilöryhmää
6) ei saa syrjiä mitään toimialaa
7) oikeudet lisenssiin tulee soveltua kaikille
8) lisenssi ei saa olla tuotekohtainen
9) lisenssi ei saa rajoittaa muita ohjelmia
10) lisenssin tulee olla teknologianeutraali.

Stamelosin ja muiden [2002] mukaan avoimien ohjelmistojen kehittäminen perustuu yksinkertaiseen ideaan: ohjelman runko on kehitetty paikallisesti yhden ohjelmoijan tai ryhmän toimesta; sen jälkeen prototyyppi järjestelmästä julkaistaan Internetissä, ja sen lähdekoodia voi lukea, muokata ja jakaa.

Stamelosin ja muiden [2002] mukaan avoimen järjestelmän evoluutio tapahtuu äärimmäisen nopeasti, paljon nopeammin kuin tyypillisessä suljetussa projektissa. Tämä on mielenkiintoinen löytö, mutta todennäköisesti ihan totta, koska kaikki kehittäjät ovat heti innoissaan parantamassa systeemiä.

3.1. Avoimien ohjelmistojen lisenssit

Avoimien ohjelmistojen lisenssit eivät ole muodoltaan yhtenäisiä. Open Source Iniativen (OSI:n) hyväksymiä lisenssejä on tällä hetkellä 69. Kaikki OSI:n hyväksymät lisenssit on tarkistettu OSI:n lisenssien tarkistusmenetelmällä, jotta taataan, että ne täyttävät avoimien ohjelmistojen voimassa olevat normit ja odotukset [OSI, 1998]. Jos lisenssistä on olemassa eri versioita, annetaan myös lisenssin versio, jota on tutkittu.

Suurin osa lisenssistä on tehty samalla kaavalla ja ne ovat hyvin samanlaisia. Suurimmasta osasta on vaikeaa löytää eroja. Tutkielmassa käydään läpi lisenssejä, jotka ovat suosittuja sekä laajalti käytettyjä. Nämä lisenssit on valittu OSI:n kategoriasta suositut ja laajalti käytetyt lisenssit. Kategoriaan kuuluu yhdeksän lisenssiä: MIT, Eclipsen julkinen lisenssi (EPL-1.0), GNU general public -lisenssi (GPL-3.0), GNU Lesser general public -lisenssi (LGPL-3.0), Apachen lisenssi (Apache-2.0), Mozillan julkinen lisenssi (MPL-2.0), BSD-2 ja -3 sekä tavallinen kehitys- ja jakelulisenssi (CDDL-1.0). Taulukosta 2 näkee näiden lisenssien ominaisuuksia. Merkintä 'x' tarkoittaa taulukossa sitä, että lisenssillä on kyseinen ominaisuus. Merkintä '-' tarkoittaa sitä, että asiasta ei ole mainintaa tai ettei ominaisuutta ole. Ominaisuuksia ovat ei-takuu ja edesvastuulauseke, onko lisenssin muokkaaminen sallittua, tarvitseeko dokumentaatioon lisätä tiedote ja päivämäärä, kun tuotetta muokkaa, sekä sisältääkö lisenssi ohjeen siitä, kuinka se otetaan käyttöön.

Lisenssin lyhenne	Ei-takuu	Edes-vastuu	Lisenssin muokkaa-minen	Tiedote muokkaa-misesta	Päivä-määrä	Käyttöön-otto-ohje
Apache-2.0	x	x	-	x	-	x
BSD-3-Clause	x	x	-	-	-	x
BSD-2-Clause	x	x	-	-	-	x
GPL-3.0	x	x	x	x	x	x
LGPL-3.0	-	-	Ei sallittu	x	-	-
MIT	x	x	-	-	-	-
MPL-2.0	x	x	x	-	-	-
CDDL-1.0	x	x	x	x	-	-
EPL-1.0	x	x	Ei sallittu	-	-	-

Taulukko 2: Avoimien ohjelmistojen lisenssien ominaisuuksia

Melkein kaikista lisensseistä löytyy ei takuuta -lauseke, jossa mainitaan, että kukaan ei vastaa siitä, että järjestelmä on toimiva ja että sitä jaetaan sellaisena kuin se on. Monista lisensseistä löytyy myös edesvastuulauseke, jossa sanotaan, ettei ketään voida syyttää järjestelmässä olevista ongelmista. Joissain lisensseissä on todella paljon tekstiä, joka antaa sellaisen kuvan, että kaikki, mitä

lisenssissä ei ole kielletty, on sallittua, koska lisenssien ei ole tarkoitus rajoittaa tuotteen elämää. Lisenssien erityisominaisuuksia yritän erotella seuraavaksi.

Apachen lisenssissä on käytetty yhtenäistä kaavaa, jota osa muistakin lisensseistä hyödyntää. Kaavassa määritellään ensin termit ja sen jälkeen asiat on lueteltu otsikoiden alle. Samanlaista kaavaa käyttää myös GPL-3.0, LGPL-3.0, MPL-2.0, CDDL-1.0 sekä EPL-1.0. Apachen lisenssissä on myös maininta siitä, ettei lisenssin omistajan nimellä saa mainostaa omaa tuotettaan. Lisenssissä ei ole mainintaa siitä, saako lisenssiä muokata omiin tarkoituksiin.

BSD-2.0 ja 3.0 ovat sama lisenssi kokonaisuudessaan, mutta BSD-2.0 on lyhyempi versio BSD-3.0:sta. BSD-2.0 on yksi lyhemmistä lisensseistä, eikä BSD-3.0 ole kovin paljoa pidempi. Niiden ero on se, että BSD-3.0:ssa määrätään, ettei ohjelmistotuotteen tekijöiden nimiä tai projektin omistajan nimeä saa käyttää mainostamaan ohjelmistotuotetta ilman kirjallista lupaa. Molemmissa on myös yhdistetty edesvastuu ja ei-takuu -lauseke. Niissä on myös ohje, kuinka lisenssi otetaan käyttöön omaan ohjelmistotuotteeseen. Lisenssit ovat myös erikoisia siitä, etteivät ne pakota jakamaan koodia tai tuotetta vapaasti.

GNU general public -lisenssissä (GPL-3.0) määrätään, että muutetut versiot täytyy merkitä muutetuiksi, jotta ongelmia ei liitetä väärin aikaisempien tekijöiden versioihin. Työn täytyy sisältää huomattavat tiedotteet kertomaan, kuka on muokannut sitä ja antaa relevantti päivämäärä. Samoin kielletään alkuperäisen materiaalin harhaan johtaminen tai vaaditaan, että muokatut versiot alkuperäisestä merkitään järkevällä tavalla eri lailla kuin alkuperäinen versio. Lisenssin lopussa on liitteenä ohjeet siitä, kuinka lisenssi liitetään omaan ohjelmaan. Lisenssi sisältää paljon tekstiä, se on yksi pisimmistä lisensseistä, ja se seuraa yhtenäistä kaavaa. Sitä ei saa käyttää muiden lisenssien kanssa, paitsi GNU Affero General Public -lisenssin kanssa, koska se on ainut yhteensopiva lisenssi.

GNU Lesser general public -lisenssin (LGPL-3.0) muokkaaminen on kiellettyä. LGPL-lisenssissä varmistetaan, että kehittäjät yrittävät taata, että ohjelmiston tärkeimmät toiminnot toimivat, vaikka ohjelmaa on muokattu. LGPL-lisenssissä on myös lauseke, jossa pyydetään pyrkimään siihen, että ohjelmisto tai sovellus suorittaa sen tehtävän, vaikkei se toimita oikeaa dataa tai funktiota.

MIT-lisenssi on kokonaisuudessaan hyvin yksinkertainen; siinä kerrotaan, kuinka tietojärjestelmää voi käyttää, kunhan vain lisää lisenssin mukana tulevan tekijänoikeustiedotteen mukaan. Lisenssissä varoitellaan myös siitä, etteivät järjestelmän tekijät ole vastuussa mistään järjestelmän puutteesta tai ongelmista, jotka järjestelmän käytöstä voi aiheutua. Lisenssi on myös yksi helpoiten

ymmärrettävistä lisensseistä. Se on myös siitä erikoinen, ettei se pakota jakamaan koodia ja ohjelmaa niin kuin suurin osa muista lisensseistä.

Mozilla Public -lisenssin versio 2.0 on tehty myös samalla yhtenäisellä kaavalla kuin muutkin pidemmät lisenssit. Lisenssi kieltää myös lisenssin nimellä tai tekijöillä mainostamisen. Siinä myös uhataan, että lisenssin myöntämät oikeudet loppuvat, jos sen ehtoja ei noudateta. Lisenssiä ei myöskään saa käyttää yhdessä muiden lisenssien kanssa.

Tavallinen kehitys- ja jakelulisenssi (CDDL-1.0) sisältää mielenkiintoisen lisäyksen edesvastuulausekkeeseen. Se ei suojaa kehittäjiä, jos kehittäjän huolimattomuuden takia joku kuolee tai vammautuu, koska laki kieltää sellaisen rajoituksen. CDDL:ssä kerrotaan, kuinka koodin pitää olla helppo löytää. Lisenssistä saa tehdä oman muokatun version, kunhan sen kertoo lisenssissä. Lisenssi myös uhkailee, että oikeudet päättyvät, jos ei noudateta ehtoja, mutta annetaan 30 päivän aika korjata asia, ennen kuin lisenssi lakkaa olemasta voimassa.

Eclipsen julkinen lisenssi (*Eclipse Public License, versio 1.0*) eli EPL-1.0 on yksi pidemmistä lisensseistä. Lisenssissä noudattaa yhtenäistä kaavaa. Lisenssissä on pykälä, jossa puhutaan ohjelmiston lisäämisestä johonkin kaupalliseen tuotteeseen. Kaupalliseen tuotteen julkaisijan täytyy puolustaa kaikkia koodin tekijöitä, jos jokin ongelma tai vika ilmenee tuotteessa.

3.2. Lisenssit ja ylläpito

Missään lukemassani lisenssissä ei mainita tietojärjestelmien ylläpitoa taikka evoluutiota. Ohjeita ylläpitoon löytyy aivan muutamasta ja niissäkin vain aivan minimaalisesti. Joissain lisensseissä kerrotaan, että jos tiedostoja muuttaa, niin niihin tulee lisätä tiedotteet siitä, että tiedostoa on muutettu. Näin on esimerkiksi Apachen lisenssissä versiossa 2.0. Jotkut lisenssit tarkentavat, että myös muokkauspäivämäärä tulee näkyä. Tämä liittyy hieman ylläpitoon ja muutoksien dokumentointiin.

Kun katsoo näitä ominaisuuksia kokonaisuutena, tuntuu oudolta, miksei kaikkiin lisensseihin ole lisätty esimerkiksi käskyä tehdä tiedotteet omista muokkauksista ja lisätä niihin päivämäärät. Tämä johtuu varmaankin siitä, että on haluttu lisenssi, joka rajoittaa mahdollisimman vähän ohjelmistotuotteen tekoprosessia ja kehitystä.

Minkäänlaista ylläpitosuunnitelmaa ei siis ole lisensseihin liitetty. Tämä on mielestäni ihan järkevää, koska lisenssien tavoite ei ole hallita niihin liitettyjä ohjelmistoja, vaan antaa niille tapa päästä kehittymään. Liiallinen neuvominen ja käskyttäminen vain vaikeuttaisi uusien kehittäjien mukaan tuloa.

3.3. Avoimien ohjelmistojen kehitys

Avoimissa ohjelmistoissa ylläpidettävyys nousee tärkeäksi ominaisuudeksi, koska heti ensimmäisen version jälkeen koko ohjelmiston kehittäminen on ylläpitotyötä. Kehittäjät vaihtuvat koko ajan, ja siksi oikeanlaiset ylläpitomenetelmät olisi tärkeää tehdä kunnolla, jotta kehitystyö olisi mahdollisimman sujuvaa.

Avoimen ohjelmiston tavoite on luoda järjestelmä, joka on tarpeellinen tai kiinnostava niille, jotka työstävät sitä, eikä täyttää mitään kaupallista tyhjiötä [Godfrey and Tu, 2000]. Useimmat avoimet ohjelmistot on luotu siksi, että kaupalliset versiot eivät ole olleet kehittäjien mielestä tarpeeksi hyviä, ja he ovat itse halunneet luoda oman version maksullisesta palvelusta.

Godfreyn ja Tun [2000] mukaan tärkeitä ominaisuuksia avoimien ohjelmistojen kehityksessä ovat aikataulutus, koodin laatu, epävakaa koodi sekä suunniteltu kehitys, testaus ja ehkäisevä ylläpito. Avoimilla ohjelmistoilla ei ole mitään kaupallista painetta tehdä valmista nopeasti, joten järjestelmää ei ole pakko julkaista ennen kuin projektin omistajat ovat tyytyväisiä järjestelmän kehittymiseen ja vakauteen. Koodin laatu ja standardit voivat vaihdella suuresti järjestelmän sisällä, kaikki eivät aina noudata standardeja tai ohjeita, vaikka ne olisivatkin olemassa. Epävakaa koodi on myös tavanomaista, koska kehittäjät haluavat jakaa tuotoksensa projektiin. Suunniteltu kehitys ja testaus sekä ennaltaehkäisevä ylläpito voivat kärsiä, kun kehittäjiä rohkaistaan aktiiviseen osallistumiseen, muttei välttämättä kärsivälliseen pohdintaan ja uudelleenorganisointiin. Koodin laatua on ylläpidetty yleensä niin, että kehittäjät käyttävät muiden käyttäjien koodia, ja näin ongelmat korjautuvat. [Godfrey and Tu, 2000]

Tietojärjestelmän kehityksessä on ratkaisevia tehtäviä, kuten testaus ja dokumentaatio, jotka jätetään kokonaan huomioimatta avoimien ohjelmistojen kehityksessä. Koodin tekijät määrittelevät vaatimukset. Vain yksityiskohtainen suunnittelu näyttää saavan jotain huomiota, mutta suurin osan työstä on ohjelmointia ja vianetsintää. [Stamelos et al., 2002]

Koponen ja Hotti [2005] tutkivat kahden suuren avoimen ohjelmiston ylläpitoprosessia ja tulivat siihen tulokseen, että ylläpitoprosessi on samanlainen kuin yleensä suljetuissa tietojärjestelmissä. Tämä väittämähän on täysin ristiriidassa Stamelosin ja muiden [2002] ja Godfreyn ja Tun [2000] kanssa. Todennäköisesti erot johtuvat siitä, että Koponen ja Hotti tutkivat suurten ja hyvin järjestelmällisten ohjelmistojen ylläpitoa, joilla oli aikataulut heidän uusien versioiden julkaisuihin, ja siksi he olivat päättäneet toteuttaa ylläpitoa kaavan mukaisesti. Tätä pitäisi kuitenkin tutkia lisää, jotta väitteestä voisi olla varma.

Ylläpidettäessä avoimia ohjelmistoja riskien arviointia ei suoriteta eikä mihinkään tarpeettoman hallinnointitehtävään kuluteta aikaa [Stamelos et al., 2002]. Tämä johtuu varmasti siitä, ettei ketään kehittäjää kiinnosta suorittaa riskien arviointia vaan on mielekkäämpää tutkia koodia ja ohjelmoida uutta. Ongelmat korjataan vasta sitten, kun ne tulevat eteen ja todennäköisesti ei paljon mietitä ennaltaehkäisevästi. Jos kyseessä olisi kaupallinen ohjelma, niin näin varmaan ei uskallettaisi toimia. Avoimilla ohjelmistoilla on periaatteessa ikuisesti resursseja, jos vain on olemassa joku kiinnostunut kehittäjä päivittämään koodia.

Stamelosin ja muiden [2002] tutkimuksessa koodin laadusta avoimissa ohjelmistoissa on outoa se, että avoimessa lähdekoodissa on korkealaatuisempaa koodia kuin voisi olettaa, sekä se, että avoimissa ohjelmistoissa on huonompilaatuista koodia kuin mitä standardissa annetaan ymmärtää. Lopulta he tulevat kuitenkin siihen tulokseen, että koodi on liian huonolaatuista ja sitä pitäisi parantaa. Heidän mukaansa rakenteellisen koodin laatu pitäisi vakiinnuttaa yhdeksi projektin tavoitteeksi. He myös haluaisivat, että koodin laatua alettaisiin valvoa ja kehittäjille annettaisiin standardeja ja käskettäisiin huomioimaan koodin rakenne, sekä aloitettaisiin takaisinmallinnus, jos tilanne alkaa näyttää huonolle.

Stamelosin ja muiden [2002] mukaan takaisinmallinnus on ainoa vaihtoehto, jotta avoimet ohjelmistot voivat selvitä suljetun kehityksen vaihtoehtona. En usko, että tilanne on näin, koska ovathan avoimet ohjelmistot vieläkin olemassa. Wang ja muut [2001] sanovat, että tärkein tekijä hyvässä ylläpidossa on yksittäisen ohjelmoijan kyvyt. Uskon, että tässä on syy siihen, mikseivät avoimet ohjelmistot ole jo kaatuneet. On todennäköisempää, että hyvät ohjelmoijat tekevät vapaasta tahdostaan uusia versioita. Toisaalta huonommilla ohjelmoijilla ei ole kiire saada koodia määräaikoihin mennessä valmiiksi, joten koodin laatu pysyy suhteellisen hyvänä.

Godfrey ja Tu [2000] huomasivat, ettei Linuxin kehitys ja kasvu toiminut niin kuin aiemmissa tutkimuksissa oli ehdotettu. Suurten tietojärjestelmien kehityksellä on tapana hidastua järjestelmän kasvaessa, mutta Linuxin kehitys oli pysynyt koko ajan jatkuvasti kasvavana.

Ylläpidon kannalta on olemassa kaksi eroa avoimilla ohjelmistoilla ja suljetuilla ohjelmistoilla: avoin ohjelmisto rohkaisee osallistumaan eikä yleensä keskity pitkän tähtäimen projektin hallinnointiin toisin kuin suljetussa ohjelmistossa, ja avoimien ohjelmistojen laatu ylläpidetään pääosin käyttäjien vianetsinnällä kuin systemaattisella testauksella. [Yu, 2006]

Yhteistyössä ja kulttuurisena toimintana ylläpito näyttäisi tuovan suuria hyötyjä järjestelmän luotettavuuteen ja suorituskykyyn [Bennet and Rajlich, 2000]. Bennet ja Rajlich [2000] väittävät, että kaikki menestyneet ja tarpeelliset tietojärjestelmät saavat käyttäjiltä pyyntöjä muutokseen ja parannuksiin. Jos esimerkiksi testauksen apuna käytetään käyttäjiä valmistusvaiheessa, niin silloin säästetään resursseja sekä kaikki ohjelman tärkeimmät funktiot saadaan todennäköisesti toimimaan paremmin. Käyttäjien uudet vaatimukset ohjelmistolta ovat onnistuneen ylläpidon vaikein osa-alue.

Lintula ja muiden [2006] mukaan avoimien ohjelmistojen tulisi tarjota enemmän dokumentaatiota niiden ylläpitoprosessistaan sekä ylläpidon työvälineistä. Tämä varmasti helpottaisi uusien tutkimusten tekoa ja antaisi lisää viitettä siihen, miten ylläpito todellisuudessa toteutetaan.

4. Yhteenveto

Avoimien ohjelmistojen lisensseissä ei ole mainittavia mainintoja ohjelmiston ylläpidosta. Olen myös tullut siihen tulokseen, ettei ylläpito-ohjeiden lisääminen lisensseihin toisi mitään muuta kuin rajoituksia avoimille ohjelmistoille. On parempi, että lisenssit pysyvät mahdollisimman yksinkertaisina, jotta ne eivät ala rajoittaa avoimien ohjelmistojen kehitystä. Ylläpito-ohjeet voidaan liittää avoimen ohjelmiston muihin dokumentteihin, jottei koodin laatu heikkenisi ja josta ne olisi mahdollisimman helppo löytää.

Tutkielman rajoitteina oli muutama asia. Kirjallisuuskatsaus on pääasiallisesti vain kolmen tieteellisten tekstien tietokannan käytön tulos. Mukana on myös muutama lähde, jotka on löydetty muista lähteistä. Mukana on paljon seminaarilähteitä, koska tieteellisiä lehtiartikkeleita ei ole aiheesta kovin montaa kirjoitettu ja julkaistu. Tarvetta uusille tutkimuksille siis edelleen on. Tutkielmassa on myös tutkittu vain yhdeksää lisenssiä, koska kaikkien (tai edes suuren osan) OSI:n hyväksymien lisenssien läpikäyminen olisi vienyt liian kauan aikaa tutkielman teon puitteissa.

Jatkossa voisi tutkia sitä, millaisia ohjeita ylläpitoon avoimien ohjelmistojen dokumenteista löytyy ja voisiko niistä luoda yhtenäisen mallin tai standardin, jota olisi helppo noudattaa. Voisi myös tutkia, miten avoin ohjelmisto kehittyy heti alusta ja tutkia milloin koodin ja arkkitehtuurin taso alkaa huonontua, ja miettiä voisiko sitä ehkäistä jotenkin. Näyttäisi myös siltä, ettei ole haastateltu avoimien järjestelmien kehittäjiä ja kuultu heidän kantojaan ylläpitoon.

Viiteluettelo

[Andrews and Lutfiyya, 2000] James H. Andrews and Hanan L. Lutfiyya, Experiences with a software maintenance project course. *IEEE Transactions on Education* **43**, 4 (November 2000), 383-388.

[Bennett and Rajlich, 2000] Keith H. Bennett and Václav T. Rajlich, Software maintenance and evolution: a roadmap. In: *Proceedings of the Conference on the Future of Software Engineering, 73-87.*

[Bhatt et al., 2004] Pankaj Bhatt, Gautam Shroff and Arun K. Misra, Dynamics of software maintenance. *ACM SIGSOFT Software Engineering Notes* **29,** 5 (September 2004), 1-5.

[Chapin et al., 2001] Ned Chapin, Joanne E. Hale, Khaled Md. Khan, Juan F. Ramil and Wui-Gee Tan, Types of software evolution and software maintenance. *Journal of Software Maintenance and Evolution: Research and Practice* **13**, 1 (January/February 2001), 3-30.

[Das et al., 2007] Sumita Das, Wayne G. Lutters and Carolyn B. Seaman, Understanding documentation value in software maintenance. In: *Proceedings of the 2007 Symposium on Computer Human Interaction for the Management of Information Technology,* poster.

[Desharnais and April, 2010] Jean-Marc Desharnais and Alain April, Software maintenance productivity and maturity. In: *Proceedings of the 11th International Conference on Product Focused Software, 121-125.*

[Eierman and Dishaw, 2007] Michael A. Eierman and Mark T. Dishaw, The process of software maintenance: a comparison of object-oriented and third-generation development languages*. Journal of Software Maintenance and Evolution: Research and Practice* **19**, 1 (January/February 2007), 33-47.

[Godfrey and Tu, 2000] Michael W. Godfrey and Qiang Tu, Evolution in open source software: a case study. In: *Proceedings of International Conference on Software Maintenance*, 2000, 131-142.

[Kajko-Mattsson, 2004] Mira Kajko-Mattsson, Problems within front-end support. *Journal of Software Maintenance and Evolution: Research and Practice* **16**, 4-5 (July - October 2004), 309-329.

[Koponen and Hotti, 2005] Timo Koponen and Virpi Hotti, Open source software maintenance process framework. In: *Proceedings of the Fifth Workshop on Open Source Software Engineering*, 7-11.

[Kozlov et al., 2008] Denis Kozlov, Jussi Koskinen, Markku Säkkinen and Jouni Markkula, Assessing maintainability change over multiple software

releases. *Journal of Software Maintenance and Evolution: Research and Practice* **20**, 1 (January/February 2008), 31-58.

[Lientz, 1983] Bennet P. Lientz, Issues in software maintenance. *ACM Computing Surveys* **15**, 3 (September 1983), 271-278.

[Lintula et al., 2006] Heli Lintula, Timo Koponen and Virpi Hotti, Exploring the maintenance process through the defect management in the open source projects - four case studies. In: *International Conference on Software Engineering*, October 2006, 53.

[Lutters and Seaman, 2007] Wayne G. Lutters and Carolyn B. Seaman, Revealing actual documentation usage in software maintenance through war stories. *Information and Software Technology* **49**, 6 (June 2007), 576-587.

[Mamone, 1994] Salvatore Mamone, The IEEE standard for software maintenance. *ACM SIGSOFT Software Engineering Notes* **19,** 1 (January 1994), 75-76.

[Mancl and Havanas, 1990] Dennis Mancl and William Havanas, A study of the impact of C++ on software maintenance. In: *Proceedings of the Conference on Software Maintenance,* 1990, 63-69.

[OSI, 1998] Open Source Iniative, http://opensource.org/licenses/index.html. Checked 22.10.2012.

[O'Neill, 1997] Don O'Neill, Software maintenance and global competitiveness. *Journal of Software Maintenance and Evolution: Research and Practice* **9**, 6 (November/December 1997), 379-399.

[Parikh, 1986] Girish Parikh, Exploring the world of software maintenance: what is software maintenance? *ACM SIGSOFT Software Engineering Notes* **11**, 2 (April 1986),49-52.

[Schneider, 1983] G. R. Eugenia Schneider, Structured software maintenance. In: *Proceedings of National Computer Conference 1983*, 137-144.

[Schneidewind, 1987] Norman F. Schneidewind, State of software maintenance. *IEEE Transactions on Software Engineering*, **SE-13**, 3 (March 1987), 303-310.

[Singer, 1998] Janice Singer, Practices of software maintenance. In: *Proceedings of the International Conference on Software Maintenance,* 1998,139-145.

[Sharon, 1996] David Sharon, Meeting the challenge of software maintenance. *IEEE Software* **13**, 1(January 1996), 122-125.

[Stamelos et al., 2002] Ioannis Stamelos, Lefteris Angelis, Apostolos Oikonomou and Georgios L. Bleris, Code quality analysis in open source software development. *Information Systems Journal* **12**, 1 (January 2002), 43-60.

[Taylor et al., 1998] M. J. Taylor, E. P. Moynihan and A. Laws, Training for software maintenance. *Journal of Software Maintenance and Evolution: Research and Practice* **10**, 6 (November/December 1998), 381-393.

[Tinnirello, 1983] Paul C. Tinnirello, Improving software maintenance attitudes. In: *Proceedings of National Computer Conference 1983*, 107-112.

[Wang et al., 2001] ShuanglinWang, Stephen R. Schach and Gillian Z. Heller, A case study in repeated maintenance. *Journal of Software Maintenance and Evolution: Research and Practice* **13**, 2 (March/April 2001), 127-141.

[Yip et al., 1994] Stephen W.L Yip, Tom Lan and Stephen K. M. Chan, A software maintenance survey. In: *Proceedings of First Asia-Pacific Conference on Software Engineering*, 1994, 70-79.

[Yu, 2006] Liguo Yu, Indirectly predicting the maintenance effort of open-source software. *Journal of Software Maintenance and Evolution: Research and Practice* **18**, 5 (September/October 2006), 311-332.

Turvallisuusarkkitehtuuri ja TOGAF-viitekehys

Elina Leino

Tiivistelmä.
Tietojärjestelmäarkkitehtuurin kriittisimmistä osa-alueista on turvallisuuteen liittyvät kysymykset. Organisaatioiden turvallisuusarkkitehtuurilla on omat metodinsa, jotka erottavat sen muista tietojärjestelmäarkkitehtuurin osa-alueista. Tutkielmassa on tavoitteena tuoda selkeästi esiin tärkeimpiä vaiheita, joita organisaation kokonais- ja turvallisuusarkkitehtuurin suunnittelun on pidettävä sisällään.

Avainsanat ja -sanonnat: TOGAF-viitekehys, kokonaisarkkitehtuuri, turvallisuusarkkitehtuuri.
CR-luokat: K.6.5

1. Johdanto

Tietojärjestelmäarkkitehtuurin yksi tärkeimmistä ja kriittisimmistä osa-alueista on turvallisuuteen liittyvät kysymykset. Ilman riittävää tietoturvaa yrityksen perustoiminnotkin ovat vaarassa. Tästä syystä on elintärkeää, että liiketoiminta-arkkitehtuurin luomisessa ja liiketoimintasuunnitelman teossa keskitytään varhain turvallisen ympäristön luomiseen.

The Open Groupin [2011] julkaisema TOGAF 9.1 -kokonaisarkkitehtuurin dokumentaatio sisältää kattavan kokonaisuuden aiheista, jotka liittyvät yrityksen tietohallintoon ja sen turvallisuuteen. Turvallisuusarkkitehtuurilla on omat metodinsa, jotka erottavat sen muista tietojärjestelmäarkkitehtuurin osa-alueista, ja se koostuu omista erotetuista näkemyksistä ja näkökulmista. Tästä syystä laaja ohjeistus turvallisuuteen liittyvistä tehtävistä on tärkeä työkalu tietojärjestelmäarkkitehdille.

Aloitan työni määrittelemällä TOGAF-viitekehyksen toiminta-ajatuksen ja yleiset alkuvalmistelut turvallisuusarkkitehtuurin suunnittelulle. Luvuissa 2, 3 ja 4 tuon esille kolme arkkitehtuurin ydinaluetta, jotka TOGAF-viitekehys pitää sisällään: liiketoiminta-arkkitehtuurin, informaatioarkkitehtuurin ja teknologia-arkkitehtuurin. Riskienhallinta ja säädökset ovat kaksi suurta aihealuetta, jotka yhdistävät kaikkia edeltäviä arkkitehtuurin osa-alueita. Nämä suuret kokonaisuudet käsittelen erillään luvussa 5. Tavoitteenani tutkielmassani on kerätä yhteen turvallisuuden näkökulmasta asioita, joita organisaation tulisi ottaa huomioon rakentaessaan toimivaa yritysympäristöä. Päälähteenä tutkielmassa on käytetty TOGAF 9.1 -kokonaisarkkitehtuurin dokumentaatio.

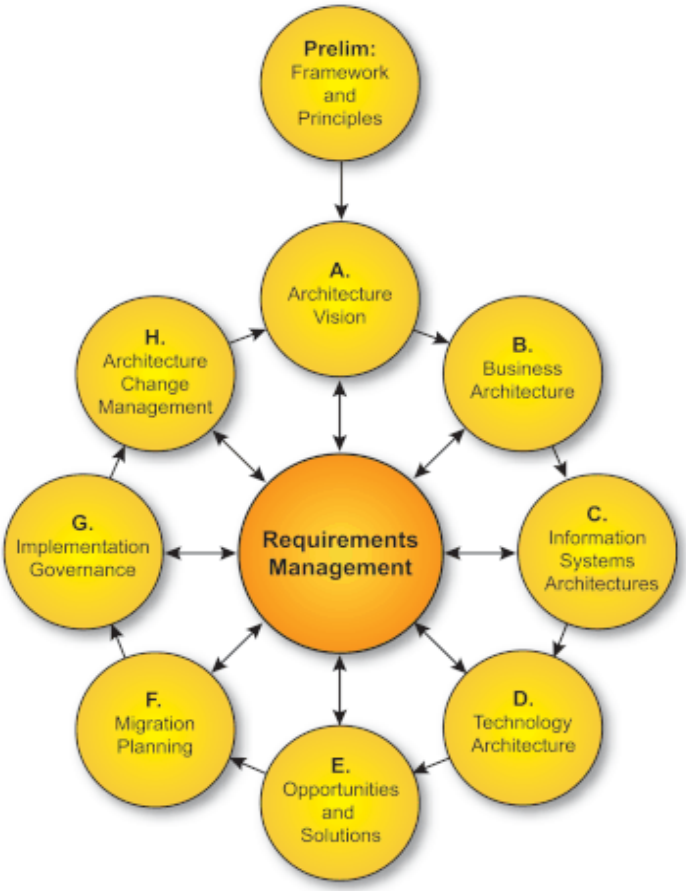
2. Turvallisuusarkkitehtuuri ja TOGAF-viitekehys

Turvallisuusarkkitehtuuri pitää sisällään niin tietoturvallisuuden kuin organisaation fyysisen turvallisuudenkin. Tietovarojen turvaamiseen liittyy paljon muutakin kuin tietokantojen ylläpito ja suojelu. Kuvassa 1 on lueteltu esimerkkejä erilaisista teknologisista ratkaisuista, joita yrityksen tietovaraston ympärillä on.

Kuva 1: Tietoturvallisuus ja tietovarojen suojelu [Valtiovarainministeriö, 2012].

TOGAF (The Open Group Architecture Framework) on kokonaisarkkitehtuurin viitekehys, joka antaa lähestymistavan kokonaisarkkitehtuurin suunnitteluun, toteutukseen ja hallintaan. Se käsittää liiketalouden, ohjelmistojen, tiedon ja teknologian näkökulmat. TOGAF-viitekehyksen pääkomponenttina on ADM (Architecture Development Method), joka kuvaa selkeästi eri komponentit kokonaisarkkitehtuurin suunnittelussa. Tässä suunnittelun metodissa on turvallisuusarkkitehtuurin muodostaminen otettu erilliseksi vaiheekseen. Turvallisuuden näkökulma on käsiteltävä erillään, koska sen infrastruktuuri on hyvin heikosti näkyvä yrityksen toiminnoille. The Open Group [2011] painottaa, että viitekehys on laajasti muokattavissa toimivaksi kokonaisuudeksi jokaiselle organisaatiolle.

Kuvassa 2 on näkyvissä TOGAF-viitekehyksen ADM-metodin vaiheet. ADM:ssa on alustava vaihe (Prelim), josta siirrytään iteratiiviseen prosessiin, joka jakaa suunnittelun kahdeksaan eri osaan. Liiketoiminta-arkkitehtuurin, informaatioarkkitehtuurin ja teknologia-arkkitehtuurin lisäksi ADM-metodi sisältää muun muassa arkkitehtuurin muutoksen hallinnan sekä mahdollisuuksien ja ratkaisujen huomioon ottamisen. ADM-metodi antaa omassa osiossaan ohjenuoran ja tekniikat kokonaisarkkitehtuurin turvallisuuspuitteiden luontiin. [The Open Group, 2011]



Kuva 2: TOGAF ADM -metodin komponentit [The Open Group, 2011].

Tyypilliset turvallisuusarkkitehtuurin kirjalliset tuotokset eli *artefaktit* ovat yrityksen tiedonhallintaa koskevat säännöstöt, kirjallinen ja julkaistu turvallisuuskäytäntö, riskianalyysi, tietovarojen omistus- ja hallussapito-oikeudet sekä tietoluokittelusäännöstöt. [The Open Group, 2011]

Turvallisuuskäytäntöjä ei ole sitoutettu pelkästään yhteen teknologiaan. Usein uusilla teknologioilla on mahdollisuus tukea turvallisuuskäytäntöjä paremmin kuin vanhoilla laitteilla tai ohjelmistoilla. Kun turvallisuuskäytännöt on kerran luotu, voidaan ne tulkita suoraan vaatimuksiksi kaikkiin arkkitehtuuriin liittyviin projekteihin. Myös turvallisuusstandardit muuttuvat usein ja niiden lisäksi uusia turvallisuuden vaatimuksia voi ilmaantua monista muistakin lähteistä. [The Open Group, 2011] Tästä syystä turvallisuusarkkitehtuurin täytyy olla joustava ja sen muuttamisen mahdollista.

Kokonaisarkkitehtuuriprosessin alussa tulisi nimetä erikseen turvallisuudesta vastaava arkkitehti. Hän pitää huolen siitä, etteivät vaatimukset ja muut arkkitehtuuria koskevat päätökset mene ristiin turvallisuusvaatimusten kanssa. Turvalli-

suusarkkitehtuuri käsittelee epänormaaleja tietovirtoja järjestelmien ja sovellusten välillä ja esittelee omat normatiiviset ja hyväksyttävät tietovirtansa. [The Open Group, 2011] Näin ollen turvallisuusarkkitehtuurista vastaavalla henkilöllä tulee olla yksityiskohtainen tietämys tietoturvallisuudesta.

IT-arkkitehti tarvitsee monipuoliset taidot hallitakseen turvallisuusarkkitehtuurin. Kokonaisarkkitehtuurin kompleksisuuden takia on myös suotavaa, että yrityksen turvallisuusarkkitehtuurille on nimetty erillinen vastuuhenkilö [The Open Group, 2011]. Varsinkin teknologia-arkkitehtuurin suunnittelussa vaaditaan paljon teknistä tietoa.

Turvallisuusarkkitehti joutuu työssään käsittelemään kahdeksaa eri aihealuetta: *todennus* (authentitication), *auktorisointi* (authorization), *auditointi* (audit), *vakuutus* (assurance), *saatavuus* (availability), *varojen suojelu* (asset protection), *hallinto* (administration) ja *riskienhallinta* (risk management). [The Open Group, 2011]

Se mitä ei pystytä mittaamaan, ei voida hallita [Jirasek, 2012]. On erittäin tärkeää, että yrityksen turvallisuustoimintojen tehokkuutta pystytään mittaamaan. Yhtenä turvallisuuden mittarina voitaisiin pitää sitä, ettei yrityksen nimeä ilmesty artikkeleihin, joissa kerrotaan yritykseen kohdistuneista tietomurroista [The Open Group, 2011]. Riittämättömästä tietoturvallisuudesta kärsiviä yrityksiä on paljon. Tämän kertoo muun muassa viikoittaiset uutiset yrityksiin ja organisaatioihin kohdistuneista tietoturvahyökkäyksistä. Tänä syksynä tavallisimmat tietoturvahyökkäykset ovat kohdistuneet organisaation asiakkaiden tietoihin, käyttäjätunnuksiin ja salasanoihin.

Arkkitehtuuriprojektin alussa on oltava selvillä kaikki johtohenkilöt, joille tulee raportoida projektin turvallisuuteen liittyvät asiat. Samalla on myös määriteltävä käytettävä turvallisuusasioiden raportointitiheys. Kaikkien turvallisuuteen liittyvien päätösten jäljittäminen on oltava mahdollista suorittaa dokumentoinnin ja kirjoitettujen raporttien perusteella. Turvallisuustoimenpiteille täytyy saada organisaation johdon tuki. Tuen saamisen tärkeys ja raportointi mainitaan TOGAF-dokumentaatiossa ADM-metodin ensimmäisessä osiossa, arkkitehtuurinäkemyksessä (architecture vision) [The Open Group, 2011]. Ilman johdon tukea ja hyväksyntää on arkkitehtuurista vastaavien henkilöiden vaikeaa saada haluamiaan muutoksia käytäntöön.

Ympäristöt, joissa järjestelmää tullaan käyttämään, on tunnistettava ja dokumentoitava ennen kuin voidaan aloittaa liiketoiminta-arkkitehtuurin suunnittelu. Näitä ympäristöjä voivat olla muun muassa yritys-, mobiili- ja ulkoilmaympäristö. [The Open Group, 2011] Esimerkiksi valittaviin turvallisuustekniikoihin vaikuttaa suuresti ympäristö, jossa järjestelmää käytetään. Verkkoympäristön tekniset ominaisuudet ja mahdolliset uhat eroavat suuresti teollisuudessa käytössä olevien järjestelmien tarpeista.

Arkkitehtuurin ensimmäisistä suunnitteluvaiheista lähtien tulee muistaa, että suunnitellut asiat voi mahdollisesti muuttua heti seuraavan iteraation myötä. Kun kaksi ensimmäistä ADM-metodin vaihetta on käyty läpi, voidaan suunnittelua jatkaa kokonaisarkkitehtuurin ydinalueisiin, joista ensimmäisenä TOGAF-viitekehys käy läpi liiketoiminta-arkkitehtuurin.

3. Liiketoiminta-arkkitehtuuri

Liiketoiminta-arkkitehtuuri ottaa huomioon yrityksen osaamisen sen omassa ympäristössään. Turvallisuuden näkökulmasta katsottuna organisaation tulisi tämän arkkitehtuurivaiheen avulla löytää päteviä ja tarpeellisia ratkaisuja ja turvallisuuskäytäntöjä. ADM-metodi painottaa, että turvallisessa liiketoiminta-arkkitehtuurissa on otettava huomioon henkilöstöresurssit ja niiden luotettavuus.

Arkkitehtuurin suunnittelussa tulisi miettiä, kenellä organisaatiosta on tietotaito ja tarve käyttää ylläpitäjän oikeuksia ja mihin järjestelmään. On selvitettävä sallitut käyttäjät, jotka toimivat ja ovat vuorovaikutuksessa prosessien kanssa. Käyttäjät eivät välttämättä ole ihmisiä, vaan ne voivat olla esimerkiksi sovelluksia. On mietittävä, ovatko organisaation ulkopuoliset tahot tarpeen järjestelmän käyttäjinä. Erilaisia käyttäjäskenaarioita on luotava, jotta kaikki tietojärjestelmän käyttäjät tunnistettaisiin ja osattaisiin ottaa huomioon. [The Open Group, 2011]

Lait ja säädökset, liiketoiminnan tavoitteet ja turvallisuusuhat ovat Jirasekin [2012] esittämät kolme päätekijää, joiden takia yritykset joutuvat tekemisiin turvallisuuden kanssa. Jos yrityksen tavoitteet ovat ristiriidassa turvallisuustavoitteiden kanssa, tavoitteet on arvioitava uudelleen [The Open Group, 2011]. Yrityksen turvallisuusvaatimusten tulee olla yhteneväiset liiketoimintatavoitteiden kanssa. Valmiita kansainvälisistä standardeista saatavia turvallisuusvaatimuksia ei ole järkevä käyttää suoraa hyväkseen. Yritysten on kannattamatonta investoida teknologiaan tai prosessiin, joka on yhtenevä turvallisuusvaatimusten kanssa, mutta ei tuo lisäarvoa itse liiketoimintaan. [Jirasek, 2012]

Esimerkkinä tavoitteiden ja turvallisuusvaatimusten yhdistämisessä on Amazon-organisaatio, jonka liiketoimintatavoitteena on pitää verkkokauppa avoinna kellon ympäri. Heidän turvallisuustavoitteenaan on täten pyrkimys pitää järjestelmä ylhäällä ja toimintakykyisenä. Järjestelmää siis suojellaan siten, että se pyritään pitämään puhtaana haittaohjelmista tai estetään mahdolliset tietoturvahyökkäykset sitä kohtaan. [Jirasek, 2012] Yksinkertaisesti ilmaistuna, yrityksen ei ole järkevää ottaa huomioon verkkokaupankäynnin turvallisuuskäytäntöjä, jos ei se verkossa kauppaa käy.

Jokaisella organisaation sidosryhmän jäsenellä on omat turvallisuuteen liittyvät huolensa ja toiveensa. Näitä huolia ei välttämättä selvästi voida ryhmitellä turvallisuuteen liittyviksi. Tästä syystä turvallisuusarkkitehti täytyisi tuoda projektiin mukaan mahdollisimman aikaisessa vaiheessa. Kaikki turvallisuuteen liittyvät päätökset täytyy olla jäljitettävissä yrityksen toimintoihin, ja ne pitäisivät olla johdettuja yrityksen riskianalyysistä. [The Open Group, 2011]

Organisaation on päätettävä, kuinka paljon käyttäjää kuormitetaan erilaisilla turvallisuuskeinoilla. Liialliset ja paljon aikaa vievät turvallisuustoiminnot voivat pelottaa asiakkaat tiehensä. Käyttäjällä tässä tapauksessa voidaan myös tarkoittaa yrityksen henkilökuntaa. [The Open Group, 2011] Monissa verkkopalveluissa voi ainoastaan kerran vuodessa vaihdettavan salasanan pakollinen käyttäminen olla asiakkaiden mielestä liikaa vaadittu. Pankkipalveluissa asiakkaat puolestaan haluavat nähdä konkreettista suojausta salasanojen ja tunnusten avulla.

Yrityksen on tunnistettava ja määriteltävä kaikki ne vastaavat eli varallisuudet, jotka ovat vaarassa, kun järjestelmään tulee toimintavirhe. Laadullisille vastaaville voi olla hankalaa määritellä tarkkaa hintaa. [The Open Group, 2011] Esimerkiksi yrityskuvan ja maineen menettämisestä voi koitua vakavia taloudellisia tappioita. Maineen kaltaisille vastaaville on kuitenkin tärkeää saada laskettua suuntaa antava arvo.

On myös määriteltävä erilaisten vastaavien omistajat. Kaikki turvallisuuteen liittyvät päätökset riippuvat luottamuksesta. Arkkitehtuurin suunnittelussa on näin ollen määriteltävä, missä luottamusta edellytetään, miten se osoitetaan ja miten siitä viestitään. Omistajana voi toimia niin organisaatio kuin työntekijäkin. [The Open Group, 2011] Henkilöiden luotettavuutta voidaan todentaa tiukemmissa tapauksissa erilaisilla taustatutkimuksilla esimerkiksi luottotiedoista tai rikosrekisteristä.

Organisaation on määriteltävä turvallisuuteen liittyvät oikeudelliset prosessit (security forensic process), jotta tietoturvaiskuja ja muita turvallisuuteen kohdistuvia hyökkäyksiä voidaan laillisin perustein käsitellä [The Open Group, 2011]. Näihin kuuluvat muun muassa varoitukset luvattomille käyttäjille järjestelmän turvatoiminnoista. Näitä prosesseja hyväksikäyttämällä yrityksellä on mahdollisuus viedä kokemansa vääryydet oikeuden käsiteltäviksi.

Pyrittäessä löytämään toimivia ratkaisuja liiketoiminta-arkkitehtuurista vastaavalla henkilöllä on oltava selkeä kuva yrityksen toiminnasta. Toisin kuin informaatio- tai teknologia-arkkitehtuuri, liiketoiminta-arkkitehtuurin suunnittelu ei vaadi suurta teknistä osaamista edes turvallisuuden näkökulmasta katsottuna.

4. Informaatioarkkitehtuuri

Turvallisuutta tarkasteltaessa informaatioarkkitehtuurin tehtävänä on tuoda yhteen fyysinen turvallisuus, tietoverkon turvallisuus ja tiedon turvaaminen. Päämääränä sillä on suojella tiedon luotettavuutta, eheyttä ja saatavuutta. [Alhabeeb *et al*., 2010]

Pilvipalveluiden suosion kasvaminen, organisaatiorakenteen häilyvämpi rajaus ja yhteistyökumppanien pääsy yrityksen sisäisiin järjestelmiin ovat tuoneet uusia haasteita tiedon turvaamiselle [Valtiovarainministeriö, 2012].

Arkkitehtuurin suunnittelussa on tunnistettava kaikki järjestelmän puolet, joiden täytyy olla muunneltavissa, kun vaihtuvuutta käytännöissä, yritysympäristössä tai pääsynhallinnassa tapahtuu. Turvallisuus tehostuu, kun turvallisuuteen liittyvät muutokset voidaan toteuttaa vähin kustannuksin. On myös toivottavaa, etteivät muutokset vaadi ohjelmakoodiin tehtäviä muokkauksia. Yleensä ohjelmistokoodiin tehtävät muutokset voivat lisätä ohjelmointivirheitä, joista koituu järjestelmään uusia haavoittuvuuksia. [The Open Group, 2011]

Tietomurtojen kontrolloinnin ja havaitsemisen takia kirjautumislokeja ja yhteyslokeja täytyy tarkistaa tasaisin väliajoin. Näitä voidaan muun muassa käyttää oikeudessa todistusaineistona. Lokien tulisi olla täsmällisiä. Kiinnijäämisen estämiseksi tietomurtajat usein pyrkivät muokkaamaan lokien tietoja. Tästä syystä lokijärjestelmiä olisi suojeltava muun muassa kryptografisin metodein. [The Open Group, 2011]

Valtionvarainministeriö [2009] on julkaissut kattavan lokiohjeen. Ohje sisältää tavoitteet lokien käsittelylle eli keräämiselle, analysoinnille, säilyttämiselle, luovuttamiselle, poistamiselle ja arkistoinnille. Lokeja tutkimalla on mahdollista tunnistaa tapahtumiin osallistuvat puolet ja varmistaa, etteivät ne kykene kiistämään osallisuuttaan tapahtumaan. Lokien avulla voidaan jäljittää tapahtumien kulkua ja havaita tunkeutumisia, poikkeumia ja järjestelmän suorituskykyongelmia. Käyttäjien oikeusturva pystytään myös varmistamaan käyttämällä suojattuja lokijärjestelmiä.

Tietovarannot, niiden omistusoikeus sekä tiedon herkkyys ja kriittisyys on tärkeä luokitella ja dokumentoida tarkasti. Tarkka ymmärrys organisaation omistamista tietovaroista auttaa toteuttamaan tiedon suojauksen mahdollisimman tehokkaasti. [The Open Group, 2011]

Järjestelmän luomat, tallentamat ja käyttämät dokumenttien herkkyys ja luokitteluryhmät on määriteltävä. Informaatioarkkitehtuuriin kuuluu myös määritellä tietovarojen hallussapidosta vastaava taho. Dokumenttien hallussapitäjä voi olla henkilö tai organisaatio. [The Open Group, 2011]

Tiedolle ja erilaisille dokumenteille on annettava käyttöikä. Vanhaa ja hyödytöntä tietoa ei kannata säilyttää tarpeettomasti, koska se kuluttaa yrityksen resursseista muun muassa kovalevytilaa. Valtuutukset ja laki voivat tuoda asiakirjojen säilyttä-

miselle tarkat aikamäärät. Suomen kirjanpitolaissa velvoitetaan säilyttämään kirjapitokirjoja vähintään kymmenen vuotta [Kirjanpitolaki 30.12.1997/1336]. Käyttöikää koskevista lakisäädöksistä pitää olla tietoisia, koska tärkeiden asiakirjojen tuhoutumisella ennen aikojaan voi olla taloudellisesti raskaat seuraukset.

Niin kuin muissakin arkkitehtuurin osa-alueissa, myös informaatioarkkitehtuurissa on analysoitava uudelleen ne tietojärjestelmän osat, joiden muokkauksen on oltava mahdollista, kun yritysympäristö, käytännöt tai pääsynhallinta muuttuvat [The Open Group, 2011]. Pääsynhallinta on aihe, joka muutokseen varautumisessa erottaa informaatioarkkitehtuurin muista arkkitehtuurin osa-alueista.

ADM-metodi painottaa käytettävän tiedon määrittelyä ja luokittelua. Se mainitsee myös potentiaalisten hyökkäysreittien analysoinnin ja lakisäädösten huomioon ottamisen tärkeänä informaatioteknologian elementtinä. Toisaalta uhkien analysointi ja säädökset ovat aiheita, jotka tulevat näkyviin jokaisessa ADM-metodin ydinarkkitehtuurikomponentissa.

5. Teknologia-arkkitehtuuri

Tietoturvallisuudesta vastaava arkkitehti ei pelkästään joudu tekemisiin sovellusten tavallisen toiminnan kanssa, vaan myös epätavallisten toimintojen, virhetilojen ja tapojen, joilla järjestelmä ja sovellus voidaan keskeyttää [The Open Group 2011]. kun IT-arkkitehti keskittyy siihen, miten järjestelmä toimii. Turvallisuudesta vastaava arkkitehti taas keskittyy siihen, miten järjestelmä voi rikkoutua.

Kokonaisarkkitehtuurin turvallisuussuunnitelma käsittää tietojärjestelmän näkökulman turvallisuutta ajatellen. Tärkeimpänä piirteenä järjestelmän turvallisuudessa on tiedon kontrolloitu käyttäminen. TOGAF tuo esille konseptin *tietoalueista* (information domains) ne, joiden avulla erilaiset tieto-objektit, käyttäjät ja turvallisuuskäytännöt jaetaan omiksi yksiköikseen. Jaottelulla pyritään selkeyttämään tietojärjestelmän ja käytäntöjen ylläpitoa. Turvallisuuden näkökulmasta tietojärjestelmä kuuluu joko paikallisen tilaajan ympäristöön (LSE) tai tietoliikenneverkkoon (CN). Turvallisuuden yleisessä arkkitehtuurisuunnitelmassa LSE ympäristöön voi kuulua kolme erillistä komponenttia. Näitä voivat olla välitysjärjestelmä, paikallinen tietoyhteysjärjestelmä ja *loppujärjestelmä* (end system). [The Open Group, 2011]

TOGAF:ssa loppujärjestelmään toteutettavat suojaukset esiintyvät kolmessa erillisessä järjestelmäpalveluiden osassa. Ne ovat käyttöjärjestelmäpalvelut (operating system services), verkkopalvelut (network services) ja järjestelmän hallinnan palvelut (system security management services). Suurin osa suojaustoteutuksista ilmentyy sovelluksissa. Laitteiston taas oletetaan suojelevan loppujärjestelmän eheyttä. [The Open Group, 2011]

Turvallisuuskontekstien luomisessa keskitytään tietoalueiden ja loppujärjestelmän resurssien hallinnan eristämiseen sekä kontrolloituun tiedon jakamiseen ja kuljettamiseen tietoalueiden välillä. Käyttöjärjestelmä eristää useita turvallisuuskonteksteja toisistaan käyttämällä laitteiston suojaustoimintoja, kuten prosessorin tilarekisteriä, eli lippurekisteriä, ja muistikuvausta (memory mapping), jotka luovat erilliset osoitteet jokaiselle kontekstille. Järjestelmän hallintaan liittyvät tietoalueiden asennus, kunnossapito ja toimeenpano. [The Open Group, 2011]

Teknologia-arkkitehtuurin alaisuuteen kuuluu metodien tunnistus, joilla resurssien kulutusta säännöstellään. Tällaisia tietotekniikan tunnettuja resursseja ovat muun muassa kovalevytila, käytettävissä oleva muistin määrä ja tiedonsiirtonopeus verkossa. [The Open Group, 2011]

TOGAF-viitekehys luottaa uudelleenkäytettävyyteen ja aikaisemmin hyväksi koettuihin teknologioihin ja tuotteisiin. Olemassa olevien turvallisuusteknologioiden ja -palveluiden uudelleenkäyttöä tulisi näin ollen harkita. Teknologia-arkkitehtuurissa on arvioitava valmiita turvallisuusohjelmistoja ja -järjestelmiä. Jos järjestelmässä on jo hyviä tapoja turvata tietoa, sitä tulisi käyttää hyväksi. [The Open Group, 2011]

Moni turvallisuusheikkouksista johtuu ohjelmointikoodin virheistä. Nämä virheet tulisi löytää mahdollisimman aikaisessa vaiheessa. Jo turvalliseksi todettua ja mitattua ohjelmointikoodin käyttöä uudelleen on järkevää harkita. [The Open Group, 2011]

Organisaation on tunnistettava ja toteutettava lieventävät toimenpiteet turvallisuuden ylläpidolle. Tässä teknologia-arkkitehtuurin vaiheessa käydään läpi klassiset turvallisuuspalvelut: todennus, auktorisointi, tiedon salassapito, tiedon eheys, kiistämättömyys (non-repudiation), vakuutus ja auditointi.

Tietojärjestelmäturvallisuudessa todennus voi yksinkertaisuudessaan tarkoittaa esimerkiksi sitä, miten käyttäjä tunnistetaan ennen pääsyn sallimista yrityksen tietojärjestelmiin. Jo perustason viranomaisvaatimuksen mukaan tunnistamisdataa ei tulisi säilyttää tietojärjestelmissä selväkielisenä. [Puolustusministeriö *et al*., 2011]

Järjestelmän käyttäjien, ylläpitäjien ja ulkopuolisten järjestelmien luotettavuutta on arvioitava. Järjestelmään on määriteltävä pienin mahdollinen pääsyoikeus, joka voidaan sallia jokaiselle entiteetille. Tämä käytännössä myös tarkoittaa sitä, että työntekijöiden taidot ja luotettavuus tulisi selvittää. [Elumalai and Kannan 2011] Entiteettien luotettavuutta käsiteltiin liiketoiminta-arkkitehtuurin yhteydessä.

Elumalai ja Kannan [2011] kertovat, ettei salasanalla todentaminen tuo riittävää suojaa mahdollisille hyökkäyksille. Tiukan turvallisuuden järjestelmiin on mahdollista kytkeä muita tunnistamisen ja todennuksen teknologioita kuten sormenjälki, kasvonpiirteet, verkkokalvon tai silmän iiriksen tunnistus, allekirjoitus tai ääni.

DNA-tunnistus on erittäin turvallinen teknologia, mutta sen käyttöönotto ei ole saanut yleistä hyväksyntää. Korkea turvallisuus on myös mahdollista luoda käyttämällä kahta erillistä todennusteknologiaa. [Elumalai and Kannan 2011]

Auktorisoinnin esimerkkinä voisi muun muassa olla se, kenellä on lupa asentaa ohjelmistoja työpisteille. Elinkeinoelämän suosituksiin kuuluu se, että turva-asetusten ja ohjelmien valtuuttamaton muokkaus on tavallisilta käyttäjiltä estetty kokonaan. [Puolustusministeriö *et al.*, 2011]

Auditoinnilla tarkoitetaan organisaation kykyä suorittaa testauksia ja arvioida omia tietoturvakäytäntöjä. Vakuutuksella taas tuodaan esiin tarve testata turvallisuutta ja sitä, että se vastaa organisaatiolle annettuja tietoturvallisuuden normeja. Turvallisuuteen liittyviä auditointeja tulisi järjestää säännöllisesti. On myös aloja, joilla kyseiset auditoinnit ovat pakollisia. [The Open Group, 2011] Mitä tärkeämpää organisaation alalle tiedon turvallisuus on, sitä useammin arviointeja tulisi tehdä. Yhteiskunnallisesti tärkeissä organisaatioissa auditoinnit ja niiden dokumentaatiot ovat tärkeässä osassa koko turvallisuusarkkitehdin työnkuvaa.

Tärkeimpänä teknologia-arkkitehtuurin vaiheena on suunnitella organisaatiolle toimiva kokonaisuus klassisista turvallisuuspalveluista. ADM-metodi ei niihin ota kantaa, mutta TOGAF-viitekehyksen dokumentaatiossa on löydettävissä *tekninen viitemalli* (Technical Reference Model), josta löytyy kattava määritelmä turvallisuuspalveluille. ADM-metodin teknologia-arkkitehtuurin komponentissa tuodaan lyhyesti esille, jo edellisissä kokonaisarkkitehtuurin osa-alueissa tulleet aiheet muutokseen varautumisesta, riskienhallinnasta ja säädöksien noudattamisesta.

6. Yhdistävät turvallisuusarkkitehtuurin aihealueet

Liiketalous-, informaatio- ja teknologia-arkkitehtuurilla TOGAF-viitekehityksen ADM-metodissa on montaa identtistä vaihetta. Mitä pidemmälle iteraatiossa mennään, sitä lyhyemmin yhteiset vaiheet mainitaan. Viitekehyksessä on nähty tarpeelliseksi toistaa asioita, jotka on tarkastettava kaikissa arkkitehtuurin osa-alueissa. Tässä luvussa tuodaan esille kaksi suurta kokonaisuutta, riskienhallinta ja säädökset, jotka vaikuttavat jokaisessa ADM -metodin komponentissa.

5.1 Riskienhallinta

Riskienhallinta on jatkuvasti muuttuva alue, jonka päämäärä on tunnistaa ennalta-ehkäisytavat ja hallintajärjestelmät, joilla käsitellä riskejä, jotka ovat kytkettyjä erityisiin toimintoihin ja arvokkaisiin varoihin [Barateiro *et al.*, 2012]. Riskienhallinta on osa jokaista arkkitehtuurin osa-aluetta, mutta liiketoiminta-arkkitehtuurissa se on näkyvin osa riskianalyysin johdosta. Kuvaan 3 on kerätty yhteen laaja riskienhallinnan aihealue.

Kuva 3: Riskien sekä sisäisen valvonnan ja riskienhallinnan tavoitteet tulosprismassa [Valtionvarainministeriö, 2009]

Esimerkki riskienhallinnan haasteista on tietojärjestelmien nopea kehittyminen. Lisää turvallisuusriskejä tulee näkyviin uusien uhkien, haavoittuvuuksien ja hyökkäysohjelmistojen muodossa. Tästä seuraa se, että muuttumattomat riskianalyysit eivät ole enää riittävä keino varautua onnettomuuksiin. Riskienhallinta on suunniteltava jatkuvaksi prosessiksi, joka reagoi nopeasti muutoksiin. [Al-Hamdani, 2006]

Yrityksen jatkuvuuden kannalta on tärkeää, että tehdään suunnitelmat onnettomuuksien. TOGAF-viitekehys jakaa järjestelmät kriittisyyden mukaan kolmeen kategoriaan. Turvallisuuskriittisillä järjestelmillä tarkoitetaan järjestelmiä, jossa virhetilasta voi vaarantua ihmishenkiä. Tehtäväkriittisissä järjestelmissä on mahdollista

menettää esimerkiksi rahaa tai markkinaosuutta. Viimeiseen ryhmään tulisi ottaa mukaan ne järjestelmät, joiden häiriötiloista koituu vain minimaalista vahinkoa. [The Open Group, 2011] On selvää, että mitä kriittisimpiä järjestelmät ovat, sitä tärkeimpiä ne yritykselle ovat. Tästä syystä näiden järjestelmien suunnitteluun, toteuttamiseen ja ylläpitoon on käytettävä aikaa.

Uhka-analyysissä kerrotaan, mitä suuren luokan uhkia voi tapahtua ja niiden todennäköisyydet. Uhkien kartoittamiseen olennaisena osana kuuluu, että määritetään myös kaikki ne turvallisuuteen liittyvät seikat, joihin projektilla ei ole mahdollista vaikuttaa. Yksi tapa tunnistaa ja kerätä mahdollisia riskejä on käyttää hyväksi arkkitehtuurikyvykkyyden kypsyystasomallia hyväksi. Mahdollisia riskejä voidaan selvittää tutkimalla sitä, mistä syystä prosessi tai organisaatio ei pääse kypsyystasomallin tasoilla etenemään. [The Open Group, 2011]

Riskien tunnistus, analysointi ja arviointi ovat tärkeitä jo ennen arkkitehtuurin käyttöönottoa, jotta uhkakuvia pystyttäisiin paremmin seuraamaan ja käsittelemään. Riskien luokittelu ja käsittely vaatii suunnittelijoilta jatkuvaa työtä ja valvontaa. Riskit luokitellaan yleensä ajan, niistä koituneiden kulujen sekä laajuuden mukaan. Alasta ja yrityksestä riippuen riskejä voidaan luokitella myös muilla tavoilla. Erilaisia luokitteluryhmiä voivat muun muassa olla liiketaloudellisiin, tietovaroihin, sovelluksiin tai teknologiaan liittyvät riskit. [The Open Group, 2011]

Riskien vaikutukset ja yleisyys on määriteltävä. TOGAF-viitekehys antaa esimerkin merkintätavasta, jolla vaikutuksia voidaan kuvata neljän kohdan asteikolla: katastrofaalinen, kriittinen, marginaalinen ja vähäpätöinen. Riskien yleisyyttä voidaan taas kuvata asteikolla tiheään toistuva, todennäköinen, satunnainen, harvinainen ja epätodennäköinen. [The Open Group, 2011]

Riskien vähentämisessä tulisi pääpainon olla usein toistuvissa tai suuren vaikutuksen tuovissa riskeissä. Tämän lisäksi myös vähemmän kriittiset riskit otetaan huomioon ja analysoidaan. ADM-metodin iteratiivisen olemuksen johdosta, riskien pienentämisen jälkeen riskien vaikutukset ja yleisyys on analysoitava ja laskettava uudelleen. Tällä tavalla voidaan varmistua, että riskien vähentämisprosessi onnistui. [The Open Group, 2011] ”Henkilöstöturvallisuustyöllä vähennetään oman henkilöstön aiheuttamaa tuottamuksellista uhkaa muun muassa ohjeistamalla, kouluttamalla, kehittämällä työmenetelmiä ja vaikuttamalla asenteisiin.” [Valtiovarainministeriö, 2008]

Virheet ja laiminlyönnit ovat uhkia, jotka vaikuttavat tiedon ja järjestelmän yhtenäisyyteen. Näitä virheitä eivät ainoastaan tuota henkilöstö, joka päivässä syöttää paljon tietoa järjestelmään, vaan myös käyttäjät, jotka tuottavat ja päivittävät tietoa. Joskus virheet ovat itsessään jo uhkia. Sellaisia ovat muun muassa ohjelmointivir-

heet, joista voi koitua järjestelmän kaatuminen. On hyvin vaikea suunnitella ohjelma, joka pystyy käsittelemään kaikki mahdolliset kirjaustyypit. Kouluttamalla tietovaroja käyttävää henkilöstöä voidaan välttyä tällaisilta ongelmilta.

Barateiro ja muut [2012] antavat mallin, jolla järjestelmän turvallisuusuhat voitaisiin luokitella. Tähän luokitteluun kuuluu kolme erillistä tekijää. Ensimmäiseksi tulee ottaa huomioon hyökkääjän tieto järjestelmästä. Voidaan olettaa, että järjestelmä on sitä haavoittuvaisempi, mitä enemmän tietoa hyökkääjällä siitä on. Esimerkiksi salasana on vaikeampi arvata, jos salasanan luonnin käytännöt eivät ole yleisesti tiedossa.

Toisena tekijänä on määriteltävä järjestelmän eri osien kriittisyys eli se miten tärkeitä osat ovat, jotka voivat joutua hyökkäyksen kohteeksi. Kolmantena tekijänä on määriteltävä yritykselle tai järjestelmälle koituvat menetykset uhkakuvan toteutuessa. Se sisältää yksityisyyden, eheyden, yrityskuvan ja rahallisen tappion. [Barateiro *et al.*, 2012]

Riskienhallinta on myös informaatioarkkitehtuurissa hyvin tärkeä osuus turvallisuuden ylläpitämisestä. TOGAF-dokumentaation [2011] mukaan organisaation on määriteltävä lähestymistavat, joilla käsitellä tunnistettuja riskejä. Riskien lieventäminen, hyväksyminen, siirtäminen ja välttäminen ovat esimerkkejä kyseisistä lähestymistavoista.

Tietohallinnan puolesta on tunnistettava saatavuuden kriittisyys ja korjaustoimenpiteet jokaiseen toimintoon. Näitä tulisi vertailla myös yrityksen jatkuvuuden tai onnettomuussuunnitelmaan kanssa. [The Open Group, 2011] Saatavuutta voidaan ajatella esimerkiksi järjestelmän toimivuutena.

ADM-metodi antaa jokaisen ydinarkkitehtuurin lopussa muistutuksen riskienhallinnasta. Jokaisen iteraation välillä tulisi pysähtyä miettimään, mitkä asiat suunnitteluprosessissa voisivat mennä vikaan. [The Open Group, 2011] Riskienhallinta on toisin sanoen turvallisuuden turvaamista. Jokaisessa turvallisuuspäätöksessä on takana pyrkimys pienentämään riskien mahdollisuutta. Voidaan siis olettaa, että liiketoiminnan riskienhallintaprosessi ja turvallisuusarkkitehtuuri suunnittelu pitävät sisällään samoja vaiheita ja yhteisiä pyrkimyksiä.

5.2 Standardien ja lakimääräysten noudattaminen

Informaatioarkkitehtuurin turvallisuuden näkökulmaan suurena vaikuttimena toimivat erilaiset standardit ja lait, jotka on huomioitava ja arvioitava arkkitehtuurin suunnittelussa. Yritykselle lainsäädäntöjen noudattaminen on pakollista, mutta standardit ovat liiketoimintatapoja, joilla on mahdollisuus tehostaa toimintaa.

Uusia turvallisuuden vaatimuksia nousee monenlaisista lähteistä, joista yhtenä on erilaiset uudet lakisääteiset määräykset ja valtuutukset [The Open Group, 2011]. Suomen Puolustusministeriö, Ulkoasiainministeriö ja Sisäasiainministeriö [2011] ovat luoneet Kansallisen turvallisuusauditointikriteeristön, jossa on otettu tietoturvallisuuden osa-alue esille. Turvallisuusauditointikriteeristö antaa vähimmäisvaatimukset sille, miten yrityksen tietovarojen luottamuksellisuutta, eheyttä ja käytettävyyttä tulisi suojata [Puolustusministeriö *et al.*, 2011]. Turvallisuusarkkitehtuurin päätarkoituksena on siis suojella organisaation tietojärjestelmän ja -varojen arvoa [The Open Group, 2011].

Kuvassa 4 on näkyvissä erilaiset Suomessa käytössä olevat liiketoiminnan vaatimukset. Yrityksillä on velvollisuutena noudattaa esimerkiksi julkisuuslakia, tietoturvallisuusasetusta sekä tekemiään kansainvälisiä sopimuksia. Näistä kaikista vaatimuksista muodostuu organisaation tietoturvallisuus.

Kuva 4: Liiketoiminnan vaatimukset [Valtiovarainministeriö, 2012]

Hyvä alku turvallisuusvaatimusten määrittelyyn on kansainväliset ISO-standardin dokumentit ISO/IEC 17799:2005 ja sen päivitetyt versiot. Standardien tarkoitus on vähentää kuluja, edistää yhteentoimivuutta ja edesauttaa innovaatioiden syntymistä. Standardit on kehittänyt alojensa asiantuntijat, joten niistä saa luotettavan ohjeistuksen tietoturvallisuuden periaatteisiin ja hyviin toimintatapoihin. [The Open Group, 2011].

7. Yhteenveto

TOGAF-viitekehys on hyvä ohjenuora yrityksille alasta ja koosta riippumatta. Se antaa hyvät ja joustavat komponentit arkkitehtuurin suunnitteluun. On muistettava, että TOGAF mahdollistaa viitekehyksen muokkaamisen jokaiselle yritykselle ja alalle sopivaksi. Voimme olettaa, että tutkielmassa lueteltuja kaikkia vaiheita ei kannata pienten yritysten toteuttaa resurssien puutteen takia. Aloittavien yritysten on muun muassa mahdotonta ottaa huomioon nykyisiä järjestelmiä, jos ei niitä ole. On aloja, joissa monien vaiheiden dokumentointi on taas hyvin nopea prosessi. Pitkään metallialalla toimineen yrityksen tietojärjestelmäarkkitehdin ei tarvitse kauaa miettiä ympäristöjä, joissa järjestelmät toimivat.

Jokaisessa ADM-metodin suunnitteluvaiheessa on yhteisiä elementtejä. Tästä syystä onkin tärkeää, että kaikki arkkitehtuurin osa-alueet luodaan tiiviissä yhteistyössä. On kuitenkin ymmärrettävä, että esimerkiksi lainsäädäntöä tarkistellessa arkkitehtuurit eivät saa kaikkea tarvitsemaansa tietoa samoista lakisäädöksistä.

Arkkitehtuurin miettiminen liiketoiminnan kannalta varmistaa nimenomaisen yrityksen toimintaa tukevat ratkaisut. Informaatioarkkitehtuurin vaiheet ovat painottuneet tietovarojen tarkkaan luokitteluun. Ilman kyseistä luokittelua, yrityksen olisi vaikea muun muassa suunnitella oma riskienhallintansa. Teknologia-arkkitehtuurin luomisessa nimensäkin mukaan tarvitaan teknistä tuntemusta. Tämä kokonaisarkkitehtuurin osa-alue pitää sisällään konkreettisen turvallisuustekniikan, joka useammille tulee ensimmäisenä mieleen turvallisuusarkkitehtuurin käsitteestä.

Tasaisin väliajoin löydetään uusia tietoturvallisuuden uhkia ja useimmat uhat tulevat tutuksi median välityksellä. Tästä syystä yrityksen turvallisuusarkkitehdin pitää olla tietoinen siitä, mitä muissa organisaatioissa on tapahtunut ja ottaa opiksi muiden tahojen tekemistä virheistä. Ilman riittävää tietoturvaa yrityksen perustoiminnatkin ovat vaarassa. Tästä syystä on elintärkeää, että kokonaisarkkitehtuurin suunnittelussa keskitytään varhain turvallisen ympäristön luomiseen.

Viiteluettelo

[Al-Hamdani, 2006] Wasim A. Al-Hamdani, Assessment of need and method of delivery for information security awareness program. In: *Proc. of 3rd Annual Conference on Information Security Curriculum Development InfoSecCD '06*, 102-108.

[Alhabeeb *et al.*, 2010] Mohammed Alhabeeb, Abdullah Almuhaideb, Phu Dung Le and Bala Srinivasan, Information Security Threats Classification Pyramid. In: *Proc*

of24th International Conference on Advanced Information Networking and Applications Workshops (2010), 208-213.

[Barateiro *et al.*, 2012] J. Barateiro, G. Antunes and J. Borbinha, Manage risks through the enterprise architecture. In: *Proc. of 45th Hawaii International Conference on System Science HICSS'12*, 3297 – 3306.

[Elumalai and Kannan, 2011] K. Elumalai and M. Kannan, Multimodal authentication for high end security, *International Journal on Computer Science & Engineering* **3,** 2 (2011), 687-692.

[Jirasek, 2012] Vladimir Jirasek, Practical application of information security models information security, *Information Security Technical Report* **17** (February 2012), 1-8.

Kirjanpitolaki 30.12.1997/1336.

[Puolustusministeriö *et al.*, 2011] Puolustusministeriö, Ulkoasiainministeriö and Sisäasiainministeriö, KATAKRI, Kansallinen turvallisuusauditointikriteeristö versio II. Saatavana: http://www.defmin.fi/files/1870/KATAKRI_versio_II.pdf Luettu 14.12.2012

[The Open Group, 2011] The Open Group, TOGAF Version 9.1. Available as http://pubs.opengroup.org/architecture/togaf9-doc/arch/. Checked 14.12.2012.

[Valtiovarainministeriö, 2012] Valtiovarainministeriö, Teknisen ICT-ympäristön tietoturvataso-ohje. Saatavana: http://www.vm.fi/vm/fi/04_julkaisut_ja_asiakirjat/01_julkaisut/05_valtionhallinnon_tietoturvallisuus/20121122Teknis/ICT_taitto.pdf. Luettu 20.12.2012.

[Valtiovarainministeriö, 2009] Valtiovarainministeriö, VAHTI 3/2009 Lokiohje. Saatavana: http://www.vm.fi/vm/fi/04_julkaisut_ja_asiakirjat/01_julkaisut/05_valtionhallinnon_tietoturvallisuus/20090511Lokioh/Vahti_3_NETTI.pdf. Luettu 20.12.2012.

[Valtiovarainministeriö, 2008] Valtiovarainministeriö, Tärkein tekijä on ihminen - henkilöstöturvallisuus osana tietoturvallisuutta. Saatavana: http://www.vm.fi/vm/fi/04_julkaisut_ja_asiakirjat/01_julkaisut/05_valtionhallinnon_tietoturvallisuus/20080218Taereki/Vahti2_08low.pdf. Luettu 20.12.2012.

Eteenpäin syöttävät neuroverkot ja vahventava oppiminen

Joel Luukka

Tiivistelmä.
Koneoppimisen avulla voidaan tuottaa toimintaansa parantavia sovelluksia. Tässä tutkielmassa perehdytään siihen, kuinka neuroverkkojen itseohjautuva oppiminen voidaan järjestää vahventavan oppimisen periaatteilla. Tutkielman kokeellisessa osuudessa vertaillaan kahden vahventavasti oppivan agentin suoriutumista yksinkertaisessa tehtävässä. Kokeen tulokset osoittavat agentin toteutuksen olevan keskeinen tekijä oppimistuloksen kannalta.

Avainsanat ja -sanonnat: eteenpäin syöttävät neuroverkot, vahventava oppiminen, SARSA-agentit.
CR-luokat: I.2.6, I.2.8, I.2.11

1. Johdanto

Koneoppiminen on osa tekoälytutkimusta, jossa pyritään tuottamaan koneita ja ohjelmia, jotka parantavat itse omaa toimintaansa. Vaikka ajatuksia älykkäistä koneista on ollut jo antiikin Kreikan aikoina, on koneoppiminen omana tutkimusalueenaan saanut alkunsa 1940-luvulla ohjelmoitavien tietokoneiden kehittämisen myötä. Neurolaskenta, joka perustuu hermosolujen oppimiskykyyn, syntyi vuonna 1943 neurotieteilijä Warren McCullochin ja matemaatikko Walter Pittsin yhteistyön tuloksena [McCulloch and Pitts, 1943].

Koneoppimisen menestyksekkäät sovellukset ovat nykyään yleisiä. Oppimiskyvyn tuottamat hyödyt näkyvät ohjelmiston tai laitteiston käyttäjälle usein välillisesti järjestelmän parempana toimintana. Esimerkiksi älypuhelimissa lähestulkoon vakiovarusteena oleva puheentunnistusominaisuus on saavuttanut nykyisen suorituskykynsä juuri koneoppimismenetelmien avulla [Mitchell, 1997]. Koneoppiminen voi myös vapauttaa ihmisresursseja tärkeämpiin tehtäviin sitä mukaan, kun koneet pystyvät ottamaan vastuulleen yhä vaativampia tehtäviä. Koneoppimismenetelmien kirjo on kuitenkin valtava, ja yksittäisen menetelmän toimivuus riippuu paljolti sovellusalueesta.

Tässä tutkielmassa tutustutaan vahventavan oppimisen (reinforcement learning) menetelmiin, erityisesti temporaalierotusmenetelmän toteuttavaan SARSA-algoritmiin, sekä neuroverkkoihin, joilla voidaan esittää jatkuvia funktioita. Lisäksi tutkitaan kuinka vahventavan oppimisen menetelmät ja neuroverkot voidaan yhdistää, ja näin tuottaa oppivia agentteja, joiden toiminta perustuu neuroverkkojen oppimiskykyyn.

Luvussa 2 tutustutaan neuroverkkoihin ja niiden ominaisuuksiin ja toimintoihin. Luvussa 3 tarkastellaan vahventavan oppimisen periaatteita ja luvussa 4 esitetään vahventavan oppimisen keskeisiä menetelmiä. Luvussa 5 tutustutaan kahteen sovellukseen, joissa vahventava oppiminen ja neuroverkot on yhdistetty, sekä tarkastellaan oppivien agenttien toteutusta. Luvussa 6 esitetään kaksi itse toteutettua oppivaa agenttia ja vertaillaan niiden suoriutumista yksinkertaisessa oppimistehtävässä. Luku 7 on yhteenveto.

2. Neuroverkot

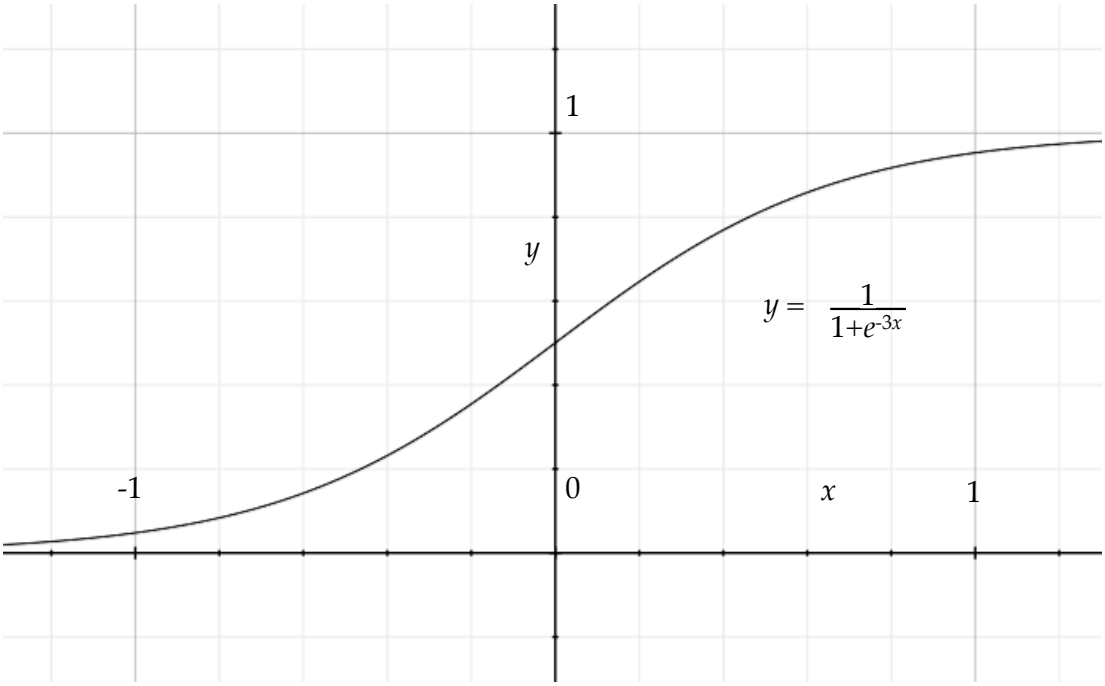
Neurolaskenta [Haykin, 2001; Rumelhart *et al.*, 1994] on koneoppimisen [Mitchell, 1997] piiriin kuuluva menetelmä, jossa yksinkertaisten laskentayksiköiden yhteistoiminnan avulla tuotetaan ratkaisuja sovellusalueen ongelmiin. Lähtökohtana neurolaskennalle on nimensä mukaisesti eläinten hermosolujen toiminta ja sen mallintaminen [Haykin, 2001]. Biologiasta omaksuttuja käsitteitä neurolaskennassa ovat neuroni, jota neurolaskennassa kutsutaan solmuksi, neuronin aktivaatio, synapsit, eli neuronien väliset yhteydet ja aktivaation ja synapsien tuottama signaali.

2.1. Yksinkertainen neuroni

Neuroni on yksinkertainen laskentayksikkö, jonka tehtävänä on ottaa vastaan signaaleja, päätellä oman aktivaation taso vastaanotettujen signaalien perusteella ja välittää omasta aktivaatiotasostaan signaali eteenpäin. Yksinkertaisimmissa malleissa aktivaatiotaso on joko päällä tai pois päältä, jolloin vastaavasti lähetettävä signaali on joko täysivahvuinen tai täysin olematon. Edistyneemmissä malleissa aktivaation laskemiseen käytetään jatkuvaa aktivaatiofunktiota, jonka arvoväli on useimmiten joko [0, 1] tai [-1, 1].

Kuvassa 1 esitetään neuroverkoille tyypillinen sigmoidiaktivaatiofunktio. Jatkuvat aktivaatiofunktiot mahdollistavat rakenteeltaan monimutkaisten

neuroverkkojen oppimisen. Kuvan 1 aktivaatiofunktion mukaan neuronin tuottaman signaalin vahvuus y määräytyy sen vastaanottamien signaalien yhteenlasketun vahvuuden x perusteella.

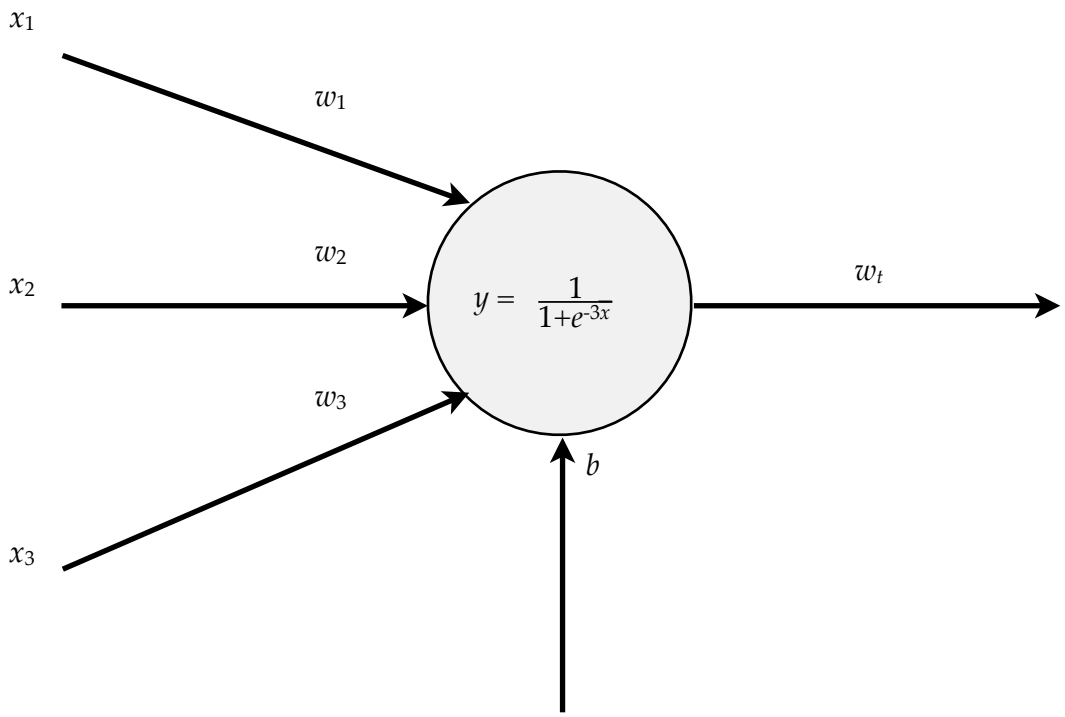


Kuva 1. Sigmoidityyppinen aktivaatiofunktio arvovälillä [0,1].

Neuronien väliset yhteydet eli synapsit kuvaavat signaalien kulkusuuntaa ja voimakkuutta. Synapsit voimistavat tehtävälle merkittävien neuronien signaaleja ja vaimentavat merkityksettömien neuronien signaaleja. Joidenkin neuronien signaalit voivat jopa olla estäviä, negatiivisia signaaleja, jolloin niiden tuottama signaali vähentää vastaanottavan neuronin aktivaatiota [Rumelhart *et al.*, 1994]. Synapsien vahvuutta kutsutaan neurolaskennan yhteydessä neuronien välisiksi painoarvoiksi, sillä neuronista lähtevä signaali koostuu lähettävän neuronin aktivaatiotasosta kerrottuna synapsin vahvuuden määrittämällä arvolla.

Neuronien välillä kulkevien signaalien lisäksi kullakin neuronilla on niin sanottu virhesyöte, joka on riippumaton muista neuroneista. Virhesyöte määrää, kuinka paljon suurempi tai pienempi neuronin vastaanottamien signaalien kokonaisvahvuuden tulee olla saavuttaakseen tietyn aktivaation tason [Rumelhart *et al.*, 1994]. Toisin sanoen, kun neuroni ei vastaanota mitään signaaleja muilta neuroneilta, määräytyy sen aktivaatiotaso virhesyötteen perusteella.

Kuvassa 2 on esitetty perusneuroni ja siihen yhteydessä olevat neuroverkon osat. Saapuvien signaalien $x_i w_i$ ja virhesyötteen b summa $x = \sum x_i w_i + b$ määrää tämän neuronin aktivaation aktivaatiofunktiossa y. Neuronin aktivaatio ja lähtevien synapsien painoarvot w_i määräävät puolestaan lähtevien signaalien vahvuuden.



Kuva 2. Perusneuronin rakenne.

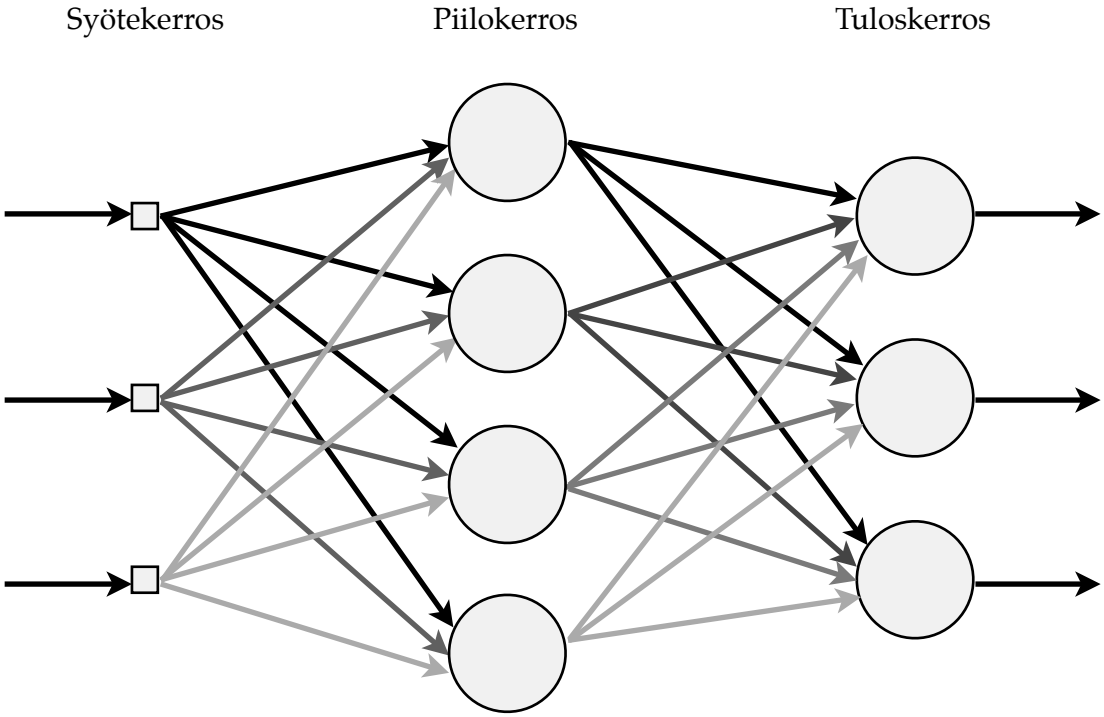
2.2. Eteenpäin syöttävä neuroverkko

Erilaisilla arkkitehtuureilla voidaan tuottaa neuroverkkoja, jotka käsittelevät tietoa eri tavoin. Neuroverkkojen sovellusalue onkin hyvin laaja. Neuroverkkoja on käytetty muun muassa luokitteluun, hahmontunnistukseen ja funktion approksimointiin [Widrow *et al*., 1994].

Tässä tutkielmassa keskitytään eteenpäin syöttäviin neuroverkkoihin (feed-forward neural network), joissa verkon syötteiden ja tulosten suhdetta voidaan helpoiten tutkia. Eteenpäin syöttävissä neuroverkoissa solmut ovat järjestetty kerroksiksi. Signaalit etenevät syötekerroksesta eteenpäin piilokerrosten kautta tuloskerrokseen. Piilokerroksia voi olla useampia kuin yksi tai niitä ei tarvitse olla

ollenkaan. Signaalit eivät kuitenkaan siirry takaisin verkossa, eivätkä ne kulje samassa kerroksessa solmujen välillä – tästä signaalien yksisuuntaisesta etenemistavasta neuroverkkotyyppi onkin saanut nimensä [Haykin, 2001].

Kuva 3 havainnollistaa yksinkertaisen eteenpäin syöttävän neuroverkon rakennetta. Verkon toiminta perustuu solmujen aktivaatioon ja aktivaation mukaisen signaalin välitykseen muunnetulla voimakkuudella seuraavaan kerrokseen. Koska solmut muodostavat itsenäisen prosessointiyksikön, jonka toiminta on laskennallisesti tehokasta, voivat eteenpäin syöttävät neuroverkot tuottaa nopeasti syötteestä prosessoidun tuloksen.



Kuva 3. Yksinkertaisen eteenpäin syöttävän neuroverkon kerroksittainen rakenne solmujen välisine yhteyksineen.

2.3. Neuroverkkojen opettaminen

Neuroverkkojen opettamiseen on useita eri lähestymistapoja. Opetustavat jaetaan usein joko valvottuun oppimiseen (supervised learning), joka on etenkin eteenpäin syöttävien neuroverkkojen opetuksessa yleisimmin käytetty opetustapa, tai valvomattomaan oppimiseen (unsupervised learning) [Haykin, 2001]. Valvotussa

oppimismenetelmässä neuroverkolle osoitetaan syöte-vastauspareja, joiden mukaan verkko pyrkii oppimaan parien osoittaman lainalaisuuden.

Valvomattomassa oppimisessa neuroverkolle esitetään tehtävästä riippumaton mittari, jonka mukaan verkko arvioi omaa suoriutumistaan datan prosessoinnissa ja korjaa itseään ilman, että oikeita vastauksia tarvitsee esittää [Haykin, 2001]. Tällaista oppimista käytetään esimerkiksi Kohosen itseorganisoituvissa kartoissa [Haykin, 2001], joissa verkko pyrkii löytämään syötedatasta yhteisiä piirteitä. Valvomattoman opetustavan etuna on se, että opetusdatan leimaamiseen ei kulu paljon resursseja. Kuitenkin myös valvomattomassa opetuksessa käytettävä data tulee kerätä siten, että se kuvaa hyvin sovellusaluetta.

Neuroverkkojen oppimiskyky perustuu solmujen välisten painoarvojen korjaamiseen haluttuun suuntaan, mikä muuttaa koko verkon dynamiikkaa. Yleisin opetusalgoritmi, jolla eteenpäin syöttäviä neuroverkkoja opetetaan, on niin sanottu takaisinlevitysalgoritmi (backpropagation algorithm), joka kuuluu valvotun oppimisen piiriin [Haykin, 2001].

Takaisinlevitysalgoritmin ensimmäisessä vaiheessa neuroverkon annetaan prosessoida syötevektori. Verkon tuottamaa tulosvektoria verrataan opetusdatan vastausvektoriin ja näiden erotuksen avulla lasketaan virhesignaali. Algoritmin toisessa vaiheessa kuljetaan verkkoa taaksepäin ja painoarvojen korjausohjeita ”levitetään” kerroksittain aina syötekerrokseen asti. Kun muistetaan, mitkä solmut kussakin kerroksessa aktivoituivat ja kuinka paljon, voidaan solmusta lähtevää synapsin painoarvoa korjata siten, että solmusta koituva virhe seuraavassa kerroksessa pienenee. Kun verkkoa opetetaan useita kertoja eri syöte-vastauspareilla, tulisi verkon antaman virhesignaalin kutistua siedettäväksi.

Huolellisesti kerätty opetusdata ja oikein toteutettu opetusprosessi johtavat parhaassa tapauksessa tilanteeseen, jossa verkko tulkitsee myös tuntemattomat, opetusdataan kuulumattomat tapaukset oikein. On kuitenkin huomattava, että monimutkaisissa sovellusalueissa osa tuntemattomista tapauksista voidaan tulkita väärin. Tämä voi johtua useasta syystä, joista eräs on ylioppiminen. Ylioppimisessa verkko tuottaa mallin, joka sopii liian hyvin opetusdataan eikä siten ole riittävän yleinen opetusjoukkoon kuulumattoman datan käsittelyyn.

3. Vahventava oppiminen

Vahventavan oppimisen lähtökohtana on ympäristön ja toimijan välinen vuorovaikutus. Sutton ja Barto [1998] tiivistävät vuorovaikutuksen avainpiirteet havaitsemiseen, toimintaan ja tavoitteeseen. Toimija havainnoi ympäristön tilaa ja valitsee tilanteeseen sopivan toiminnon, jonka tulee johtaa kohti toimijan tavoitetta. Vahventava oppiminen on varsinaisesti oppimisongelma, jonka ratkaisemiseen on tuotettu useita menetelmiä. Näitä menetelmiä kutsutaan vahventavan oppimisen menetelmiksi.

Toisin kuin valvotussa ja valvomattomassa opetustavassa, vahventavassa oppimisessa ympäristöä ei välttämättä tunneta. Sen sijaan toimijan on kokeiltava eri toimintojen vaikutuksia ympäristön tilaan ja muodostettava kokemuksen perusteella kokonaiskuva ympäristöstä [Sutton and Barto, 1998]. Näin vahventava oppiminen on mahdollista tilanteissa, joissa valvottuun tai valvomattomaan oppimiseen tarvittavaa tietoa ei ole saatavilla. Vahventavan oppimisen etuna on myös se, että toimijat pystyvät sopeutumaan ympäristön muutoksiin. Ympäristön ei tarvitse olla staattinen aina samoin toimiva kokonaisuus, vaan se voi olla dynaaminen, ajan myötä muuttuva.

Selvennetään vahventavan oppimisen lähtökohtia esimerkin avulla. Pöytätenniksen pelaajat pyrkivät saavuttamaan pelissä tilanteen, jossa he pystyvät vaivatta vastaanottamaan vastustajan lyönnit ja vaikuttamaan vastustajan lyönteihin. Pelaaja, joka ”hallitsee” peliä, pystyy lyömään sellaisia lyöntejä, joihin vastustaja ei pysty vastaamaan. Jotta pelaaja pystyisi hallitsemaan peliä tiettyä pelaajaa vastaan, on tämän opeteltava vastustajan tapa vastata erilaisiin lyönteihin – minkälaisiin lyönteihin tämä vastaa vaivatta, mitkä ovat vastustajalle vaikeita lyöntejä ja minkälainen pelityyli puolestaan antaa vastustajalle vahvemman aseman pelissä.

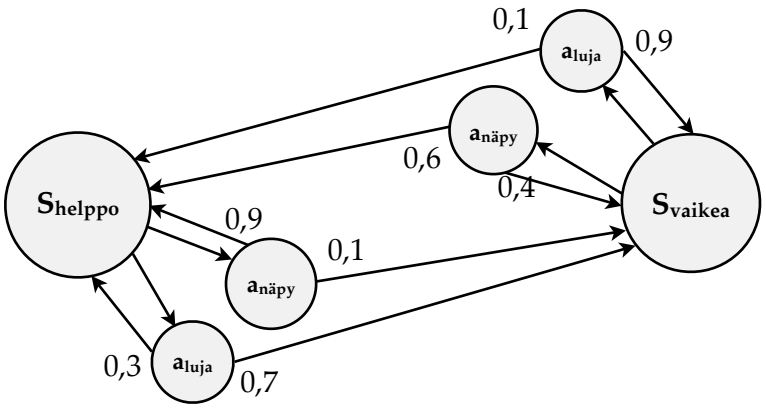
Tässä esimerkissä vastustajan reaktio voidaan ymmärtää ympäristön tilana. Toimijan mahdolliset toiminnot ovat tämän osaamat erilaiset lyönnit. Tavoitteena on voittaa peli ja pelin aikainen palkinto on se, kuinka paljon pelaaja hallitsee peliä. Pelin edetessä pelaaja oppii, kuinka tulisi pelata, jotta päätyisi hallitsevaan asemaan pelissä, mikä johtaa useimmissa tapauksissa voittoon. Koska molempien pelaajien voidaan olettaa opettelevan vastustajansa pelityyliä samaan aikaan pelissä, tulevat vastustajan reaktiot erilaisiin lyönteihin muut-

tumaan pelin edetessä. Vahventavan oppimisen näkökulmasta tämä ei ole ongelma, sillä jo opittujen toimintatapojen tilalle voidaan aina oppia parempia toimintatapoja.

Ympäristöllä viitataan erilaisiin asioihin sovellusalueesta riippuen. Ympäristö ei aina tarkoita sananmukaisesti toimijaa ympäröivää fyysistä tilaa, vaan se voi kattaa tätä suppeamman alueen, kuten robotin raajojen asentoa mittaavat komponentit. Toisaalta ympäristö voi kattaa laajemman kokonaisuuden, kuten tehdashallin kaikki lämpötila- ja kosteusmittarit ja ilmanvaihtokoneiston toiminnan tilan. Jos ympäristö ymmärretään hyvin suppeaksi alueeksi, voi tulla sekaannusta myös siinä, mikä on ympäristön ja toimijan välinen raja.

Eräs vahventavan oppimisen tärkeimmistä analysointityökaluista on Markovin päätöksentekoprosessi (Markov decision process), joka kuvaa ympäristöä ja siinä toimimisen mahdollisuuksia ja seurauksia [Sutton and Barto, 1998]. Prosessi on joukko tiloja ja tila-toimintopareja, joiden avulla voidaan muodostaa suunnattu graafi. Markovin päätöksentekoprosessin avulla sovellusalueen ympäristö voidaan kuvata selkeästi. Prosessi esittää myös ympäristön ja toimijan välisen eron – ympäristö on se, johon kuuluu jokaista tilaa kuvaavat parametrit, ja toimija on se prosessointiyksikkö, joka kulkee graafin sivuja pitkin tilasta toiseen.

Kuva 4 esittää yksinkertaisen graafin Markovin päätöksentekoprosessista pöytätennisesimerkin pohjalta. Kuvassa suuret ympyrät s_{helppo} ja s_{vaikea} kuvaavat ympäristön, tässä vastustajan reaktion, mahdollisia tiloja. Tila s_{helppo} tarkoittaa sitä, että vastustajan on helppo vastata pelaajan syöttöihin ja s_{vaikea} tarkoittaa vastustajan olevan ahtaalla. Pelaajalla on mahdollisuus valita joko hitaan näpylyönnin $a_{näpy}$ ja lujan iskun a_{luja} välillä. Pienistä toimintoympyröistä lähteviin nuoliin on yhdistetty todennäköisyys seuraavasta ympäristön tilasta.



Kuva 4. Markovin päätöksentekoprosessi pöytätennisesimerkistä.

Ympäristön ja toimijan välinen vuorovaikutus on perusperiaatteiltaan yksinkertaista. Toimija havainnoi ympäristön nykytilaa ja päättelee sisäisen mallin perusteella toiminnon, joka tuottaa parhaan palkkion. Kun toiminto on suoritettu, toimija vastaanottaa ympäristöltä palkkion ja korjaa sisäistä malliaan sen perusteella. Tämän jälkeen toimija havainnoi jälleen ympäristöään, päättelee parhaan toiminnon ja niin edelleen. Jotta toimijan sisäinen malli korjaantuisi vastaamaan ympäristön todellista olemusta, on toimijan välillä valittava toiminto, joka ei tuota sisäisen mallin mukaan parasta tulosta [Sutton and Barto, 1998]. Kokemuksen kartuttua toimijan sisäisestä mallista poistuvat väärät oletukset ja ympäristön muuttuessa se pystyy sopeutumaan muutoksiin. Toimijan valintoja ohjaa sille asetettu tavoite ja sen pohjalta lasketut palkkiot kustakin valinnasta kussakin tilassa. Oppijan tavoitteena on maksimoida kokonaispalkkio, joka lasketaan kaikkien palkkioiden summana.

4. Vahventavan oppimisen menetelmiä

Vahventavan oppimisen toteuttamiseen on useita lähestymistapoja ja niiden variaatiota. Tässä esitellään kolme perustavanlaatuista menetelmää, jotka kaikki ratkaisevat oppimisongelman kokonaisuudessaan. Nämä menetelmät ovat dynaaminen ohjelmointi, Monte Carlo -menetelmä ja temporaalierotuksen menetelmä [Sutton and Barto, 1998]. Sutton ja Barto osoittavat, että näitä menetelmiä on mahdollista yhdistää eri tavoin, mikä tuottaa lähes rajattoman määrän erilaisia toteutuksia. Menetelmäkuvaukset perustuvat Sutton ja Barton kirjaan ellei toisin ilmoiteta.

4.1. Dynaaminen ohjelmointi

Dynaamisella ohjelmoinnilla haetaan ympäristön kuvauksen perusteella optimaalista toimintaohjetta, jonka mukaan toimessaan oppiva agentti maksimoi keräämänsä palkkion. Jotta toimintaohje voidaan löytää, tarvitaan ympäristöstä tarkka ja totuudenmukainen kuvaus. Useimmissa tapauksissa ympäristöstä ei ole saatavilla niin tarkkaa kuvausta, että dynaamista ohjelmointia voitaisiin käytännön sovelluksessa toteuttaa, mutta sen periaatteet ovat tärkeässä asemassa vahventavassa oppimisessa. Moni muu opetusmenetelmä pyrkiikin imitoimaan dynaamista ohjelmointia ilman, että tarkkaa ympäristön kuvausta tarvittaisiin.

Dynaamisen ohjelmoinnin menetelmä perustuu toimintaohjeille laskettuihin arvofunktioihin ja paremman toimintaohjeen etsimiseen niiden avulla. Tila-arvofunktio $V^{\pi}(s)$ kuvaa tilalle s odotetun palkkion, kun noudatetaan toimintaohjetta π. Toimintoarvofunktio $Q^{\pi}(s, a)$ puolestaan kuvaa tilalle s odotetun palkkion, kun seuraavaksi tehdään valinta a toimintaohjetta vastaan, ja sen jälkeen noudatetaan toimintaohjetta π. Kun toimintoarvofunktion Q arvo tilassa s ja toiminnolla a on suurempi tai yhtä suuri kuin tila-arvofunktion V määräämä arvo tilassa s, voidaan vaihtoehtoisen toimintaohjeen olettaa olevan parempi tai vähintään yhtä hyvä kuin nykyisen toimintaohjeen. Nykyinen toimintaohje voidaan siis vaihtaa paremmaksi havaittuun ja laskea sille uusi tila-arvofunktio V. Tätä vähittäistä parantelua jatkettaessa saavutaan lähemmäksi optimaalista toimintamallia.

Arvofunktioiden V ja Q arvoalue määräytyy ympäristön tarjoaman suurimman ja pienimmän palkkion mukaan ja arvofunktioiden päivityksessä käytettyjen alennuskertoimien mukaan. Alennuskertoimien vaikutus on kuitenkin pieni, joten käytännössä arvofunktioiden arvoalue on jotakuinkin pienimmästä palkkiosta suurimpaan.

Esimerkkimme pöytätenniksen pelaaja on hankkinut tulevan vastustajansa viimeisimmistä peleistä videotallenteet ja seuraa niiden avulla vastustajansa pelityyliä. Tallenteiden perusteella pelaaja muodostaa mielikuvan vastustajan reaktioista tiettyihin lyönteihin ja suunnittelee omaa pelistrategiaansa tämän perusteella. Dynaamisen ohjelmoinnin mukaan pelaajan tulisi harkita kussakin tilanteessa s, kuinka vastustaja reagoisi vaihtoehtoiseen lyöntiin a ja kuinka peli

jatkuisi tämän jälkeen, jos jatketaan pelistrategian π mukaan. Jos pelaajan mielikuvissa hänen asemansa paranee, tai pysyy samana vaihtoehtoisen lyönnin *a* jälkeen, voidaan pelistrategiaan omaksua vaihtoehtoisen lyönnin *a* käyttö pelitilanteessa *s*, johon sitä sovitettiin.

4.2. Monte Carlo -menetelmä

Monte Carlo -menetelmä approksimoi tila-arvofunktiota *V* hyödyntäen esimerkkitapauksia. Esimerkkitapaukset kuvaavat Markovin graafia pitkin kuljettuja polkuja ja niistä kerättyjä kokonaispalkkioita. Monte Carlo -menetelmä laskee keskiarvon palkkiosta kullekin tilalle *s* sen mukaan, kuinka esimerkkitapaukset kulkevat tilan *s* läpi ja asettaa tämän keskiarvon tila- arvofunktion *V* arvoksi tilassa *s*. Koska arvofunktio lasketaan kokonaispalkkion mukaan, tulee esimerkkitapauksien selkeästi päättyä johonkin – joko päätöstilaan tai muuten määritettyyn päätössääntöön. Tämä sulkee pois ajonaikaisen oppimisen esimerkiksi jatkuvassa oppimisongelmassa. Tarkkaa kuvausta ympäristöstä ei kuitenkaan tarvita, sillä arvofunktioita tarkennetaan jatkuvasti niiden ympäristön elementtien osalta, joita esimerkkitapaukset kuvaavat. Monte Carlo -oppimisen ajatuksena on se, että kun esimerkkitapausten määrä lähestyy ääretöntä, tulee ympäristön kaikissa tiloissa harjoiteltua jokaista mahdollista toimintoa äärettömän monta kertaa, jolloin täydellinen ymmärrys ympäristöstä ja sen lainalaisuuksista voidaan saavuttaa.

Kun ympäristö esittää Monte Carlo -agentille tilan *s*, agentti valitsee toiminnon perustuen omaan tila-arvofunktioonsa *V*(*s*). Jos agentti valitsee joka kerta parhaan mahdollisen vaihtoehdon, jonka arvofunktio *V* osoittaa, tulisi kullekin tilalle harjoitettua vain tiettyä toimintoa. Tiedon hyödyntämisen ja uuden tiedon etsimisen välisen tasapainon löytäminen on yksi Monte Carlo -menetelmän keskeisiä ongelmia. Maksimoidakseen palkkionsa agentin on tehtävä valintoja, joiden se tietää antavan suuren palkkion. Toisaalta oppiakseen uusia, mahdollisesti parempia toimintamalleja agentin tulisi rohkeasti kokeilla uusia vaihtoehtoja. Jotta kaikki vaihtoehdot päivittyisivät, ja näin Markovin graafin kaikki sivut tarkentuisivat kohti todellisia arvojaan, tulee agentin silloin tällöin valita muu kuin paras vaihtoehto jokaisessa ympäristön tilassa. Yleinen valintamenetelmä on niin sanottu ε-ahne menetelmä, jonka mukaan agentti valitsee ahneen vaihtoehdon, eli parhaan vaihtoehdon arvofunktion *V* mukaan, paitsi

pienellä todennäköisyydellä ε, jolloin agentti valitsee jonkin muun vaihtoehdon.

Pöytätennisesimerkin pelaaja seuraa tallenteelta omaa suoriutumistaan viimeisimmässä pelissään ja koettaa tämän perusteella pohtia, kuinka tätä vastustajaa vastaan tulisi pelata seuraavassa ottelussa. Monte Carlo -menetelmän mukaan pelaaja arvioi kussakin ympäristön tilassa kullekin lyönnille odotusarvon tulevasta pelin hallittavuudesta sen perusteella, kuinka tallenteella eri tiloissa on eri lyöntejä ilmentynyt. On mahdollista, että joitakin tila-lyöntipareja ei esiinny ollenkaan tai niitä esiintyy niin vähän, että parin odotusarvo jää miltei satunnaiseksi arvioksi. Huonosti määritetty odotusarvo voidaan korjata vasta seuraavan ottelun jälkeen, jos odotusarvoa vastaavia tila-lyöntipareja tuolloin esiintyy tallenteella. Pelaaja voi pelin aikana suorittaa muutamia kokeilevia lyöntejä, jotta saa paremman kokonaiskäsityksen vastustajansa reaktioista. Seuraavan pelin strategia voidaan kuitenkin approksimoida uusien odotusarvojen perusteella vasta pelin päätyttyä, mikäli tila-lyöntipareille lasketut odotusarvot antavat strategian muutokselle aihetta.

4.3. Temporaalierotusmenetelmät

Temporaalierotusmenetelmät laajentavat Monte Carlo -menetelmää siten, että toimija pystyy päivittämään arvofunktiotaan jokaisella aika-askeleella sen sijaan, että päivitys suoritettaisiin vasta esimerkkitapauksen päätyttyä. Tämä mahdollistaa jatkuvan, toiminnan aikaisen oppimisen, joka soveltuu hyvin tapauksiin, joissa ei ole selkeää lopetusehtoa. Temporaalierotusta merkitään usein TD(λ), jossa parametri λ osoittaa, kuinka monta ylimääräistä edellistä tilaa päivittävät arvofunktiota viimeisimmän askeleen perusteella. Koska TD(λ) toimii periaatteessa samoin kuin Monte Carlo -menetelmä, kun λ on esimerkkitapauksen pituus vähennettynä yhdellä, käsitellään tässä temporaalierotusta parametrilla nolla, jolloin arvofunktiota päivittää vain edellinen tila s.

Yleisimpiä temporaalierotukseen perustuvia algoritmeja ovat Q-oppiminen, SARSA-oppiminen ja toimija-kriitikko -menetelmät. Tutkielmassa esitetään Q-oppimisen ja SARSA-oppimisen algoritmit, sillä nämä ovat yksinkertaisia ratkaisuja temporaalierotusoppimisen toteuttamiseksi. Toimija-kriitikko -menetelmät vaativat kahden arvofunktion ylläpitoa, kun taas sekä Q- että SARSA-oppimisalgoritmit toimivat yhden arvofunktion mukaan.

Q-oppimisen perusideana on, että toimija päivittää arvofunktiotaan toimin-

nasta suoraan saavutetun palkkion r perusteella ja odotetun tulevaisuuden palkkion perusteella. Tällöin toimija valitsee varsinaisen toiminnon a_1 esimerkiksi ε-ahneella menetelmällä ja käyttää Q-arvon päivitykseen tarvittavan toiminnon a_2 valintaan puhtaasti ahnetta menetelmää. Seuraava suoritettava toiminto valitaan kuitenkin arvofunktion päivitykseen valitun toiminnon a_2 sijaan ε-ahneella menetelmällä. Q-oppiminen siis päivittää toimintoarvofunktionsa nykyisestä toimintamallista riippumatta parhaan mahdollisen tulevaisuuden palkkioapproksimaation mukaisesti. Tästä syystä Q-oppiminen ei ole *toiminta-mallin mukainen* menetelmä. Q-algoritmin päivityskaava on

$$Q(s_1, a_1) \leftarrow Q(s_1, a_1) + \alpha[\, r + \gamma \max_a Q(s_2, a) - Q(s_1, a_1) \,]$$

Päivityskaavan muuttujat α ja γ määrittävät oppimisnopeuden ja tulevaisuu-denodotuksiin luottamisen asteen, tässä järjestyksessä. Kaavassa valitaan paras mahdollinen vaihtoehto kaikista vaihtoehdoista uudessa tilassa, jonka avulla arvofunktio päivitetään.

Esimerkkimme pöytätennisottelun aikana pelaaja toimii pelistrategiansa mukaan suorittaen kokeilevia lyöntejä aina silloin tällöin. Kunkin lyöntinsä jälkeen pelaaja arvioi lyöntiä siitä saadun suoranaisen peliedun ja optimistisen tulevaisuuden arvion perusteella: ”jos hyvin käy, tulee pelietuni olemaan x, joten voin olettaa aiemmassa tilassa valitun lyönnin johtavan likimain etuun x.”

SARSA-algoritmi on Q-oppimisesta johdettu toimintamallin mukainen algoritmi. SARSA-algoritmi on saanut nimensä niistä arvoista, joita algoritmi käyttää toimintoarvofunktion päivittämiseen. Nykyinen tila s_1 ja siinä valittu toiminto a_1 muodostavat parin toimintoarvofunktiolle Q. Toiminnon a_1 jälkeen tarkastellaan saatua palautetta r ja arvofunktion Q arvoa uudessa tilassa s_2 seuraavaksi valitulla toiminnolla a_2. Algoritmi käyttää toimintojen a valitsemiseen yhtä tiettyä mallia, esimerkiksi ε-ahnetta menetelmää, joten algoritmin sanotaan olevan toimintamallin mukainen menetelmä. SARSA-algoritmin päivityskaava on

$$Q(s_1, a_1) \leftarrow Q(s_1, a_1) + \alpha[\, r + \gamma Q(s_2, a_2) - Q(s_1, a_1) \,]$$

SARSA-algoritmin päivityskaavassa on sama periaate kuin Q-algoritmin

päivityskaavassa. Kaava valitsee tulevaisuuden odotuksen ilmaisevan arvon sen perusteella, mitä agentti tulee tekemään.

Esimerkiksi pöytätenniksen pelaaja valitsee lyönnin ε-ahneella menetelmällä ja arvioi lyöntiään sen perusteella, kuinka paljon pelinhallintaetua tämä lyönnistään saavuttaa ja kuinka paljon hänen seuraavan lyöntinsä odotetaan antavan pelinhallintaetua. Pelaajan ajatusmalli olisi siis realistisempi kuin optimistisen Q-oppijan ajatusmalli: ”koska tässä tilassa valittu lyönti johti etuun x, tulee minun olettaa kyseisessä tilassa kyseisen lyönnin johtavan aina likimain etuun x”.

Q-algoritmin ja SARSA-algoritmin ero on siis hyvin pieni. Algoritmit eroavat vain siinä, millä tila-toimintoparin (s_2, a) arvolla tila-toimintoparin (s_1, a_1) arvo päivitetään. Q-algoritmi valitsee päivitykseen parhaan kaikista vaihtoehdoista riippumatta siitä, mitä tekee seuraavaksi, kun taas SARSA-algoritmi valitsee sen vaihtoehdon, jonka aikoo seuraavaksi suorittaa.

5. Neuroverkon opettaminen vahventavan oppimisen avulla

Koska vahventavan oppimisen teoria ei ota kantaa oppimisympäristön tai oppivan agentin käyttämiin rakenteisiin, on neuroverkkojen opettamisessa vahventavan oppimisen avulla kysymys neuroverkon sovittamisessa oppivan agentin rooliin. Neuroverkko onkin luonnollinen valinta arvofunktion V tai Q approksimoijaksi, sillä neuroverkkoja on helppo opettaa takaisinlevitysalgoritmilla kohti toivottua arvoa. Tämän lisäksi tulee myös pohtia, kuinka ympäristö, ympäristön tila, agentin toiminto ja ympäristöltä saatu palkkio sovitetaan järjestelmään, joka opettaa neuroverkkoa.

Käytännön sovelluksia neuroverkon ja vahventavan oppimisen yhteensovittamisesta ovat tehneet esimerkiksi Tesauro [1995], jonka neuroverkko toimii tila-arvofunktiona, ja ten Hagen ja Kröse [2003], joiden neuroverkko toimii toimintoarvofunktiona.

5.1. Neuroverkko tila-arvofunktiona

Tesauro [1995] on käyttänyt neuroverkkoa backgammon-pelin lopputuloksen ennustamiseen missä tahansa pelitilanteessa ja agentin siirron valitsemiseen ennusteiden perusteella. Tesauron agentti pystyi valitsemaan parhaan mahdollisen siirron approksimoimalla jokaisen tulevan pelitilanteen, jotka voidaan

saavuttaa noppien osoittaman siirtomäärän avulla. Tässä neuroverkon syötteeksi on annettu kussakin peliruudussa olevien pelimerkkien määrä ja tieto siitä, kuuluuko merkki agentille itselleen vai vastustajalle. Neuroverkko laskee suhteellisen odotusarvon kullekin pelin lopputulokselle, joita tässä yksinkertaistetussa backgammon-pelissä ovat voitto, häviö, voitto gammonilla ja häviö gammonilla.

Tesauron järjestelmässä TD-Gammonin, eli agentin, arvofunktiona toimii neuroverkko. Neuroverkko toimii tila-arvofunktiona V, sillä se kuvaa tilan siitä odotettuun hyötyyn. Neuroverkko ei siis ota syötteenään toimintoa kuvaavia parametreja tai palauta toiminnoille hyvyysarvoa tuloksenaan. Järjestelmän ympäristö on pelilauta ja ympäristön tila on kuvaus pelimerkeistä kullakin pelilaudan ruudulla. Ympäristön tila on koodattu siten, että neuroverkko pystyy käsittelemään sen syötteenään. Agentti valitsee toiminnokseen siirron, joka johtaa parhaaseen arvioon pelin päättymisestä uuteen ympäristön tilaan perustuen. Palkkio jaetaan vasta pelin päätyttyä, vahvistaen viimeisille siirroille tulevaisuuden ennustetta, johon peli päättyi. Peliä pelattaessa useita kertoja, Tesauron kokeessa satoja tuhansia kertoja, tulevaisuuden odotukset ”levittyvät” arvofunktion päivitysten myötä viimeisistä siirroista aina pelin aloitustilanteeseen asti [Tesauro, 1995].

TD-Gammon on erinomainen esimerkki neuroverkkojen ja vahventavan oppimisen yhdistämisestä, sillä TD-Gammon on yltänyt mestarisarjaan niin ihmis- kuin konepelaajien joukossa. TD-Gammon on jopa muuttanut mestaripelaajien vakiintunutta pelistrategiaa, sillä jotkut sen erikoisista valinnoista johtavat parempiin tuloksiin kuin mihin aiemmin vallitsevat strategiat ovat johtaneet [Tesauro, 1995].

Tesauron tapa liittää neuroverkon syöte ympäristön tilaan on intuitiivisesti järkevä: agentin toiminta perustuu ympäristön nykytilaan. Toiminnon valinta voisi kuitenkin olla tehokkaampi, sillä monimutkaisempiin ongelmiin, joissa mahdollisten tilojen kirjo on laajempi, voi jokaisen tilan arviointi viedä liikaa aikaa. Tehokkaampi järjestelmä voidaan saavuttaa käyttämällä neuroverkkoa kuvaamaan toimintoarvofunktiota Q, kunhan neuroverkko ei laajene liikaa lisätyn toiminnallisuuden vuoksi. Laajempi neuroverkko voi kuluttaa yhtä paljon aikaa kuin pienemmän neuroverkon ajaminen useampaan kertaan. Tämä Bellmanin [1961] esittämä ulottuvuuksien kirous on tunnettu ongelma, joka pätee neuroverkkoihin solmujen määrän kasvattamisen yhteydessä.

5.2. Neuroverkko toimintoarvofunktiona

Ten Hagenin ja Krösen [2003] neuro-Q-agentti oppii hakeutumaan kohti ennalta määrättyä suoraa ja kulkemaan sitä pitkin saavutettuaan suoran. Robotti havaitsee oman etäisyytensä suorasta ja kulkusuuntansa ja päättelee niiden perusteella tarvittavan kulkusuunnan muutoksen. Kunkin aika-askeleen palkkio lasketaan robotin ja suoran etäisyyden perusteella. Ympäristö ten Hagenin ja Krösen tutkimuksessa on robotin fyysinen ympäristö ja robotin tavoitesuora yhdessä. Keskeisiä ympäristön ominaisuuksia ovat suora, robotin sijainti ja kulkusuunta.

Neuro-Q-agentti käyttää neuroverkkoaan toimintoarvofunktiona. Samoin kuin TD-Gammonin tapauksessa, ympäristön tila koodataan neuroverkolle syötteeksi. Toisin kuin TD-Gammonissa, neuro-Q-agentin tulossolmu osoittaa tulevan tilan haluttavuuden sijaan kulkusuunnan muutoksen määrän, joka johtaa parhaaseen mahdolliseen tilaan tulevaisuudessa. Koska tulossolmun arvo ja siitä johdettu ohjauskäsky ovat reaalilukuja, jolla osoitetaan paras mahdollinen käännöskulma, olisi kaikkien mahdollisten toimintojen approksimointi erikseen mahdotonta, koska mahdollisten toimintojen määrä jatkuvalla välillä on ääretön.

Tila-toimintopariin pohjautuva lähestymistapa olisi mahdollista toteuttaa myös diskreeteissä ympäristöissä, kuten backgammon-pelin yhteydessä, mutta järjestelmän rakenne olisi huomattavasti Tesauron järjestelmää monimutkaisempi. Ei ole kuitenkaan selvää, onko ten Hagenin ja Krösen lähestymistapa oppimistuloksen kannalta parempi esimerkiksi backgammon-pelin yhteydessä kuin TD-Gammon toteutus.

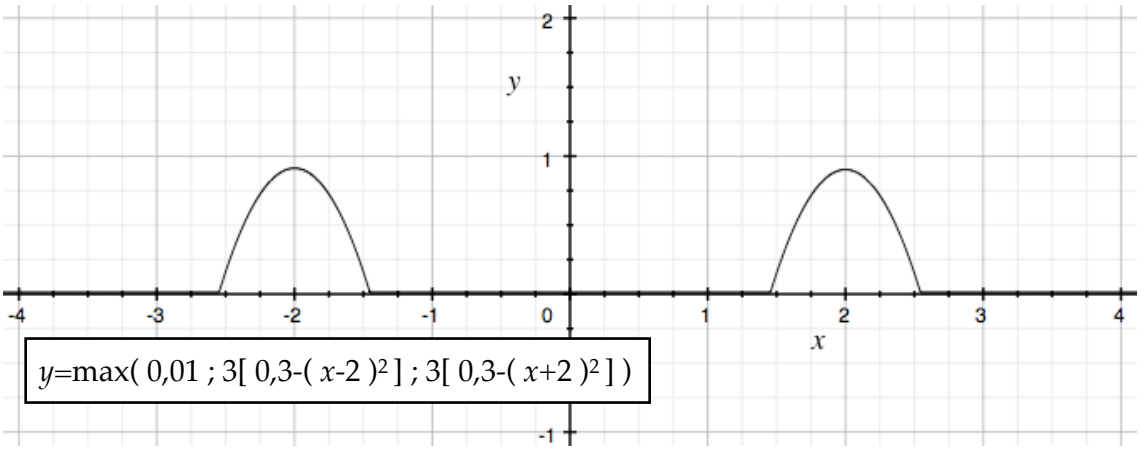
6. SARSA-agenttien toteutusten vertailu

Tutkimuksen kokeellisessa osuudessa perehdytään oppimiseroihin agenteilla, joista ensimmäinen käyttää parhaan toiminnan päättelyyn tila-arvofunktiota V ja toinen toimintoarvofunktiota Q. Kokeessa pyritään selvittämään alkeellisella tasolla, onko toiminnan päättelytavoissa eroa oppimisnopeuden tai jatkuvan oppimiskyvyn kannalta.

Koe suoritettiin Mac OS X 10.6.8 -käyttöjärjestelmässä GNU Compiler Collection (GCC) C++ -standardikirjastoa ja kääntäjää käyttäen (versio 4.2). Kokeen suoritukseen käytetyn tietokoneen suoritinnopeus on 2,53 GHz ja käyttömuistin kokonaismäärä 4 Gt.

6.1. Koejärjestelyt

Oppimisympäristö on yksiulotteinen lukujana arvovälillä [-10, 10]. Ympäristön tila on mikä tahansa reaaliluku tältä väliltä. Agentille annetaan palkkio sen lähestyessä pisteitä -2 ja 2 siten, että pieni alue pisteiden lähellä tuottaa palkkion, joka kasvaa lähestyessä palkkiopistettä. Laajennettu palkkioalue takaa sen, että agentti löytää halutun alueen ympäristöstä. Liian kapea alue voi johtaa tilanteeseen, jossa agentti ei saa koskaan palkkiota. Palkkiofunktio on esitetty kuvassa 5. Ympäristön tilat, jotka tuottavat palkkion arvolta 0,85 tai enemmän ovat tavoitetiloja, jotka päättävät opetusajon.



Kuva 5. Palkkiofunktio palkkiopisteiden -2 ja 2 lähistöllä.

Ansaitakseen palkkion agentin tulee päättää, siirtyykö se vasemmalle vai oikealle nykyisestä paikastaan. Tila-arvofunktioon nojautuva agentti valitsee parhaan toiminnon punnitsemalla kummankin vaihtoehdon tuottaman tilan arvon arvofunktion V avulla. Tätä agenttia voidaan kutsua harkitsevaksi agentiksi. Toimintoarvofunktioon nojautuva agentti puolestaan valitsee toiminnon sen perusteella, kumman se olettaa olevan parempi toimintoarvofunktion Q perusteella. Tätä agenttia voidaan kutsua spontaaniksi agentiksi.

Agenttien neuroverkot ovat yksinkertaisia eteenpäin syöttäviä verkkoja, joilla

on yksi solmu syötekerroksessa, kahdeksan solmua piilokerroksessa ja tuloskerroksessa yksi solmu harkitsevan agentin tapauksessa ja kaksi solmua spontaanin agentin tapauksessa. Kumpikin agentti siirtyy 0,2 yksikköä lukujanalla vasemmalle kohti pistettä -10 tai oikealle kohti pistettä 10 valitusta toiminnosta riippuen. Jos agentti yrittää siirtyä ympäristön alueen [-10, 10] ulkopuolelle, ei agenttia tällöin siirretä ollenkaan. Agenttien arvofunktioiden päivitykseen käytetään SARSA-algoritmia, jonka tuottama uusi arvo opetetaan neuroverkolle takaisinlevitysalgoritmilla.

Kokeessa mitataan opetusajojen määrä ja oppimistuloksen laatu. Oppimistuloksen laatu on määritelty siten, että nolla osoittaa, ettei agentti toimintaohjeellaan aina löydä palkkiotilaa. Laatua yksi olevat tulokset osoittavat, että yksi palkkiopiste on opittu ja laatu kaksi tarkoittaa molempien palkkiopisteiden oppimista. Sopivan toimintaohjeen löytäminen osoittaa, että agentti ymmärtää, kuinka ympäristön palkkio määräytyy. Oppimisnopeuden lisäksi tutkitaan, tuottaako jatkuva oppiminen parempia toimintaohjeita tai haitallista poisoppimista ja siten toimintaohjeen laadun heikkenemistä.

Varsinaisia mittaussuureita kokeessa ovat opetusajokertojen määrä ja oppimistuloksen keskiarvoinen laatu 250 opetusajon jaksoissa. Lisäksi kokeessa tallennetaan agenttien neuroverkkojen jokaisen tulossolmun tuottamat arvot välillä [-10, 10] siten, että tulossolmujen arvo mitataan 0,2 yksikön välein.

Kokeessa agentin sisäisen mallin esittävän neuroverkon painoarvot alustetaan satunnaisesti välille [0; 0,2]. SARSA-algoritmin päivityskaavan oppimisnopeutta hillitsevä askelparametri α on vakio 0,5 ja tulevaisuudenodotuksiin nojaamista säätelevä alennuskerroin γ on 0,2. Kunkin opetusajon askelraja on 60 askelta ennen kuin ajo lopetetaan kesken. Muussa tapauksessa ajo lopetetaan, kun agentti pääsee päättävään tilaan. Opetusajo aloitetaan aina satunnaisesta pisteestä. Opetusajoja yhdessä koesuorituksessa on yhteensä 60000. Agenttien kokeilukerroin alkaa arvolla 1 ja se päivitetään kunkin opetusajon jälkeen kaavalla

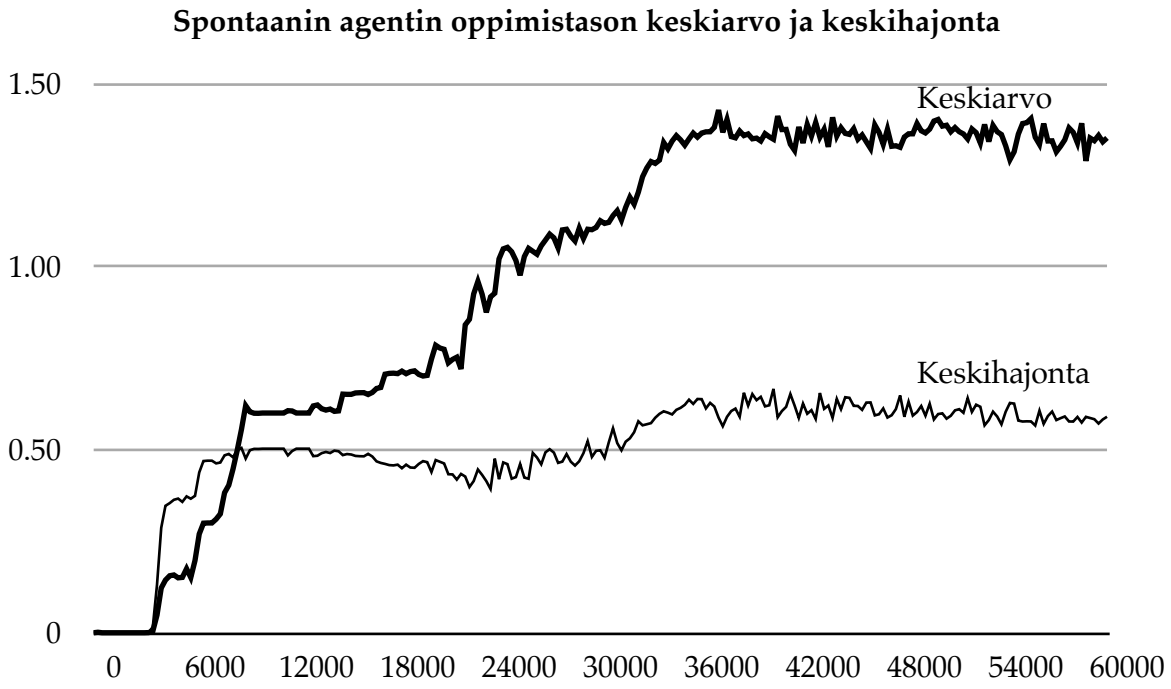
$$\varepsilon_{t+1} = 0{,}6\varepsilon_t + 0{,}2,$$

jossa t on aika-askel.

6.2. Mittaustulokset

Kokeita suoritettiin kummallakin agentilla kaksikymmentä kertaa. Kokeiden tuloksista laskettiin oppimistuloksen keskiarvo ja keskihajonta 250 opetusajon välein. Spontaani agentti suoriutui oppimistehtävästä selvästi paremmin kuin harkitseva agentti. Agenttien oppimistuloksen laadun maksimit ovat vastaavasti 1,43 ja 0,83.

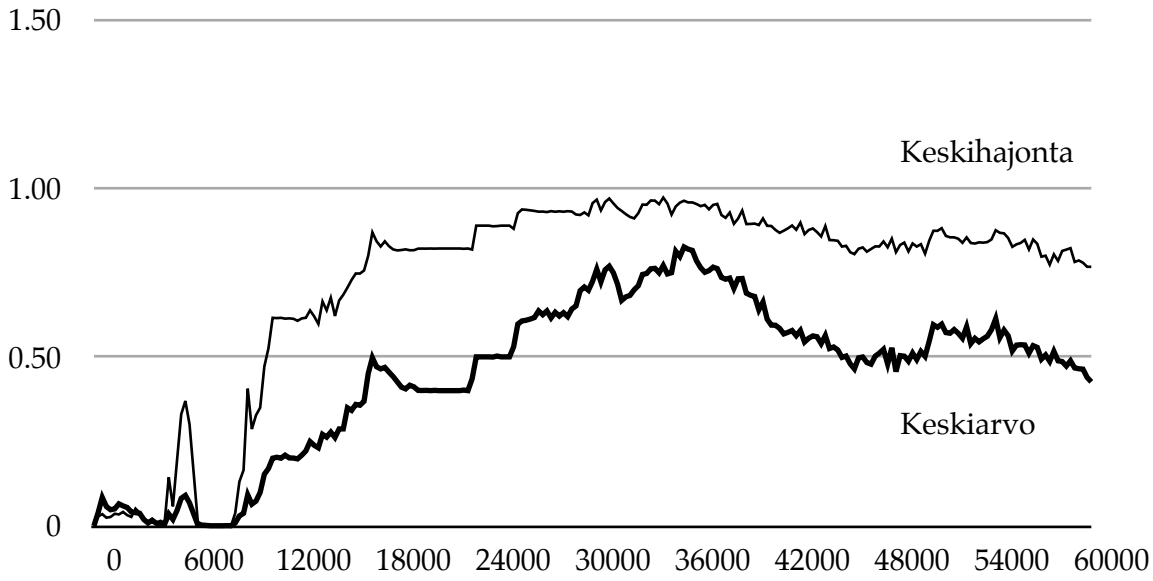
Spontaanin agentin oppimistuloksen laadun kehitys ilmenee vaihtelevantasoisena kasvuna aina opetusajolle 37000 asti, jonka jälkeen oppimistuloksen laatu jatkuu melko tasaisena, keskiarvon vaihdellessa arvojen 1,43 ja 1,29 välillä. Oppimistuloksen laadun keskihajonta opetusajon 37000 jälkeen tasaantuu likimain arvoon 0,60. Oppimistulosta voidaan pitää hyvänä, sillä agentti oppii ainakin toisen palkkiopisteen sijainnin lähes jokaisessa kokeessa. Vain kahdessa kokeessa agentin oppimisen laatu jäi alle laatuarvon yksi kokeen päätyttyä. Spontaanin agentin oppimistason kehitys on esitetty kuvassa 6. Kuvassa Y-akselin arvot vastaavat oppimislaatuja, eli opittujen palkkiopisteiden määrää.



Kuva 6. Spontaanin agentin keskimääräinen kehitys kahdessakymmenessä koesuorituksessa.

Harkitsevan agentin keskiarvoisen oppimistuloksen laadun kehitys ilmenee maltillisena kasvuna opetuskierrokselle 35500 asti, jolloin oppimistuloksen laatu on noin 0,83. Tämän jälkeen oppimisen laatu tekee pienen notkahduksen, jonka jälkeen laatu laskee maltillisesti. Oppimistuloksen laadun keskihajonta on jopa 0,97, kun se saavuttaa maksiminsa noin opetusajolla 35500. Tämän jälkeen keskihajonta laskee hitaasti arvoon 0,77 opetusajoon 60000 mennessä. Oppimistulos harkitsevalla agentilla on heikko, sillä jopa 13 koetta harkitsevalla agentilla päätyi hyvin lähelle oppimistulosta nolla ja kolmessa muussa kokeessa oppimistulos jäi alle oppimistuloksen yksi. Harkitsevan agentin oppimistason kehitys on esitetty kuvassa 7.

Harkitsevan agentin oppimistason keskiarvo ja keskihajonta

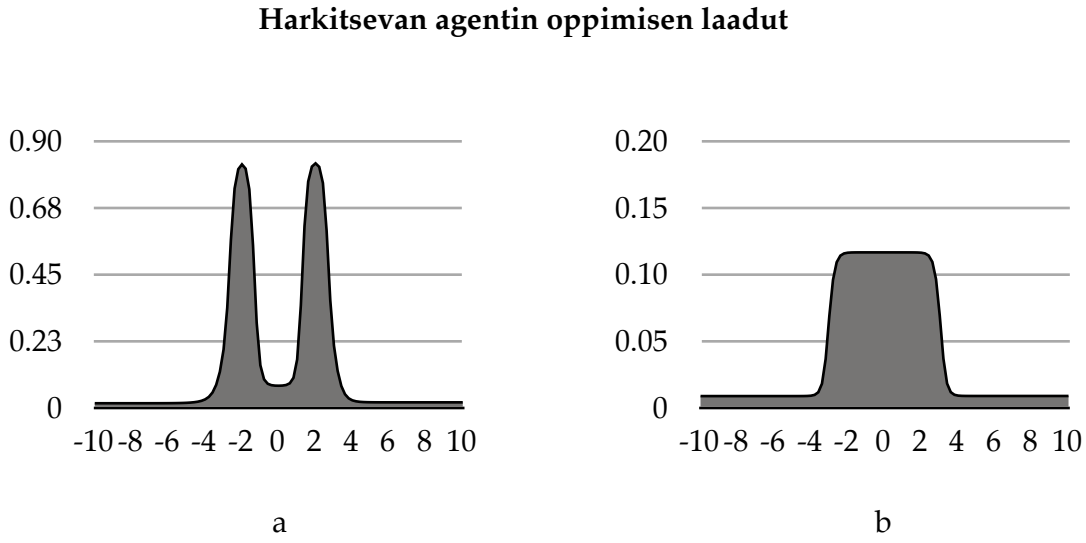


Kuva 7. Harkitsevan agentin oppimistason keskimääräinen kehitys kahdessakymmenessä koesuorituksessa.

Harkitsevan agentin heikko menestys voidaan selittää vaatimuksilla, joita agentin arvofunktion tulee täyttää. Saavuttaakseen täydellisen toimintaohjeen tulee harkitsevan agentin muokata neuroverkkonsa mukailemaan tarkasti palkkiofunktion ominaisuuksia – neuroverkon esittämän arvofunktion huippujen tulee olla täsmälleen opetusajon päättävien palkintopisteiden kohdalla ja välillä]-∞, -2] arvofunktion tulee olla aidosti kasvava. Välillä [2, ∞[arvofunktion

tulee vastaavasti olla aidosti vähenevä. Viimeisen kahden kriteerin täyttyminen osoittautui neuroverkolle haasteeksi, sillä 11 koesuorituksessa agentti osoitti ymmärtävänsä palkkioiden sijaitsevan pisteiden -2 ja 2 lähistöllä, mutta toimintaohje muuttui virheelliseksi useimmiten välillä]-∞, -6] ja [6, ∞[. Yhdeksässä muussa tapauksessa agentti oppi tulkitsemaan koko välin [-2, 2] yhdeksi palkkiopisteeksi.

Kuvassa 8 esitetään esimerkit harkitsevan agentin onnistuneen ja epäonnistuneen oppimisen tuloksista koesuorituksen päätyttyä 60000 opetusajon jälkeen. Oppimisen onnistuessa agentti hakeutuu pisteisiin -2 ja 2. Epäonnistuneessa oppimisessa agentti hakeutuu jonnekin pisteiden -2 ja 2 välille.

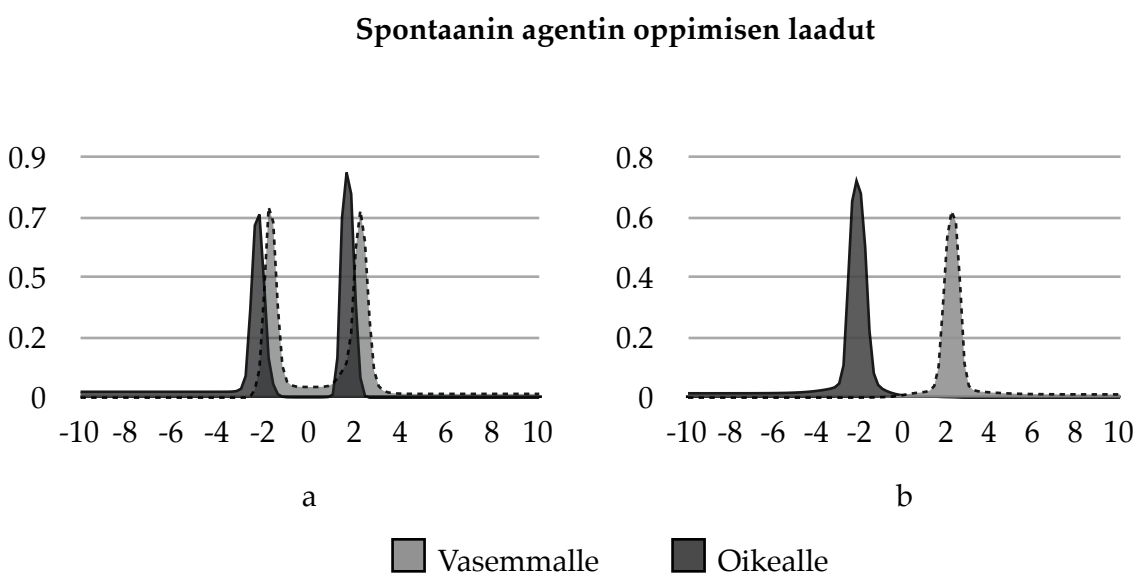


Kuva 8. Harkitsevan agentin onnistuneen (a) ja epäonnistuneen (b) oppimisen tuottamat arvofunktiot.

Spontaanin agentin tapauksessa täydellisen toimintaohjeen saavuttaminen ei aseta korkeita vaatimuksia neuroverkolle. Vaatimukset ovat, että välillä]-∞, -2] toimintoa ”oikealle” vastaava tulossolmu aktivoituu voimakkaimmin ja vastaavasti välillä [2, ∞[toimintoa ”vasemmalle” vastaava tulossolmu aktivoituu voimakkaimmin. Lisäksi tulossolmujen esittämien funktioiden tulee leikata pisteissä -2 ja 2. Koska arvofunktion ei tarvitse mukailla yhtä tarkasti palkkiofunktion ominaisuuksia kuin harkitsevan agentin tapauksessa, on spontaanin agentin helpompi muodostaa oikeanlainen toimintaohje. Spontaani agentti

saavutti seitsemässä kokeessa täydellisen toimintaohjeen ja löysi näiden lisäksi 11 kokeessa varmasti toisen palkkiopisteen mistä tahansa aloituspisteestä.

Kuvassa 9 annetaan esimerkit spontaanin agentin onnistuneen ja epäonnistuneen oppimisen tuloksista koesuorituksen päätyttyä. Onnistuneessa oppimisessa toimintojen arvot leikkaavat pisteiden -2 ja 2 lähistöllä ja agentti hakeutuu kohti palkkiopisteitä, kun se sijaitsee pisteen -2 vasemmalla puolella tai pisteen 2 oikealla puolella. Epäonnistuneessa oppimisessa agentti hakeutuu jonnekin palkkiopisteiden väliin.



Kuva 9. Spontaanin agentin onnistuneen (a) ja epäonnistuneen (b) oppimisen tuottamat arvofunktiot.

6.3. Oppimisnopeus

Oppimisnopeus määräytyy pitkälti oikeanlaisen osaamisen määrittävien vaatimusten mukaan. Kuten aiemmin oppimistuloksista pääteltiin, harkitsevan agentin korkeammat vaatimukset johtavat heikompaan oppimiseen kuin spontaanin agentin oppiminen, jolla on matalammat vaatimukset. Kokeen 60000 opetusajokierroksella spontaani agentti oppi huomattavasti harkitsevaa agenttia nopeammin.

Harkitsevan agentin kohdalla on otettava huomioon oppimisen suhteellinen hidastuminen. Kun arvofunktio lähestyy todellisuutta kuvaavaa funktiota,

arvofunktioon kohdistuvat korjausparametrit pienenevät. Korjausparametrien pieneneminen johtaa siihen, että agentti saattaa näennäisesti juuttua huonoon toimintamalliin moneksikin opetuskierrokseksi, vaikka oppimisen kannalta sopivia toimintoja kokeiltaisiinkin useaan kertaan. Kokeen opetusajojen rajoituksen puitteissa harkitsevat agentit oppivat heikosti, mutta 11 kokeessa harkitseva agentti olisi voinut oppia täydellisen toimintaohjeen, mikäli opetusajomäärä olisi ollut suurempi ja agentin oppiminen olisi jatkunut riittävästi hidastumisesta huolimatta.

Spontaanit agentit oppivat nopeasti verrattuna harkitsevien agenttien kankeaan oppimiseen. Jo 24000 opetusajokierroksen jälkeen agentit oppivat keskimäärin toisen palkkiopisteistä. Oppimisen huipun spontaanit agentit saavuttivat opetusajokierroksella 37000, minkä jälkeen oppiminen vaikuttaisi pysähtyvän. Oppiminen saavuttaa siis päätepisteensä alle 40000 opetusajokierroksen.

6.4. Jatkuva oppiminen

Kuten kuvien 6 ja 7 perusteella huomattiin, spontaanin agentin oppiminen koetilanteen muuttumattomassa ympäristössä pysähtyy noin ajokierroksen 37000 jälkeen. Harkitsevan agentin oppiminen puolestaan kääntyy ”poisoppimiseksi” kierroksen 35500 jälkeen, vaikkakaan oppiminen ei ole kovin hyvää tasoa siihen mennessä saavuttanut. Harkitsevan agentin oppiminen jää ikään kuin kesken. Pahimmassa tapauksessa oppimisen taso jää matalaksi, eikä koskaan nouse.

Agenttien opetusmenetelmän toimintaperiaatteen mukaan agenteilla on kyky oppia ympäristön ominaisuudet uudelleen, mikäli palkkiopisteiden paikka ja niistä saatava palkkio muuttuisivatkin. Ainoa rajoitus jatkuvalle oppimiselle muuttuvassa ympäristössä syntyy agenttien käyttämän neuroverkon rakenteesta – koska neuroverkon rakennetta ei voi muuttaa, voivat agentit oppia vain rajatun määrän ympäristön ominaisuuksia. Oppimisnopeus vaikuttaa suuresti agenttien suorituskykyyn muuttuvissa ympäristöissä. Nopeasti muuttuvassa ympäristössä harkitseva agentti saavuttaisi tuskin koskaan tyydyttävää oppimistasoa. Sen sijaan spontaani agentti pystyisi toimimaan muuttuvassa ympäristössä, sillä se oppii nopeammin.

Jatkuvassa oppimisessa nousee esiin toimintamallin ehdottoman noudat-

tamisen ja uuden tiedon etsimisen tasapainottelu. Saavuttaakseen suurimman mahdollisen palkkion tulee agentin noudattaa toimintamallia, jonka on oppinut. Toisaalta muuttuvassa ympäristössä voi syntyä uusia mahdollisuuksia, jotka voivat tuottaa suuremman palkkion. Tällöin agentin tulisi kokeilla vaihtoehtoja, jotka se sillä hetkellä kokisikin huonommiksi vaihtoehdoiksi. Agentin kokeilukerroin on hyvä pitää suurempana kuin nolla jos ympäristö on muuttuva. Tällöin ympäristön muutokset on mahdollista havaita nopeammin kuin tilanteessa, jossa kokeilukerrointa päivitetään vähitellen kohti arvoa nolla.

7. Yhteenveto

Vahventava oppiminen on tehokas tapa opettaa oppiville agenteille taitoja, joita on vaikea kuvata täsmällisesti. Periaatteessa toivottua toimintaa ei tarvitse kuvata muuten kuin esittämällä palkkio hyvästä suorituksesta ja rangaistus huonosta suorituksesta. Opetuslogiikan siirto koneelle itselleen vapauttaa järjestelmän suunnittelijoiden resursseja ja mahdollistaa yhä monimutkaisempien tehtävien teettämisen koneellisesti. Vahventava oppiminen vaatii ympäristöltä ja agentilta vähän, mikä mahdollistaa vahventavan oppimisen menetelmien soveltamisen lukuisilla eri alueilla.

Neuroverkkojen yhdistäminen vahventavan oppimisen menetelmiin funktion approksimoijana laajentaa edelleen vahventavan oppimisen menetelmien soveltamismahdollisuuksia jatkuviin oppimisympäristöihin. Vahventavan oppimisen ja neuroverkkojen yhdistelmä vaikuttaisikin olevan sopiva ratkaisu moneen koneoppimista vaativaan sovellukseen.

Vahventavasti oppivan neuroagentin ominaisuudet riippuvat paljolti vahventavan oppimisen menetelmän toteutuksesta ja agentin neuroverkon rakenteesta. Tutkielman kokeellinen osuus osoitti, että eri sovelluksessa hyväksi koetut agenttien toteutukset eivät välttämättä tuota hyvää oppimistulosta, kun agentti siirretään erilaiseen ympäristöön. Vahventavasti oppivasta neuroagentista tuskin tuotetaan yleispätevää oppivaa agenttia, sillä yksin vahventavan oppimisen menetelmiä ja niiden variaatioita on valtavasti.

Viiteluettelo

[Bellman, 1961] Richard E. Bellman, *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton, NJ, 1961.

[ten Hagen and Kröse, 2003] Stephan ten Hagen and Ben Kröse, Neural Q-learning. *Neural Comput. Appl.* **12** (2003), 81-88.

[Haykin, 2001] Simon Haykin, *Neural Networks: A Comprehensive Foundation*. Pearson Education (Singapore) Pte. Ltd, 2001.

[McCulloch and Pitts] Warren McCulloch and Walter Pitts, A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* **5** (1943), 115-133.

[Mitchell, 1997] Tom M. Mitchell, *Machine Learning*. McGraw-Hill, New York, NY, 1997.

[Rumelhart *et al*., 1994] David E. Rumelhart, Bernard Widrow and Michael A. Lehr, The basic ideas in neural networks. *Communications of the ACM* **37**, 3 (1994), 87-92.

[Sutton and Barto, 1998] Richard S. Sutton and Andrew G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[Tesauro, 1995] Gerald Tesauro, Temporal difference learning and TD-Gammon. *Commun. ACM* **38**, 3 (1995), 58-68.

[Widrow *et al*., 1994] Bernard Widrow, David E. Rumelhart and Michael A. Lehr, Neural networks: applications in industry, business and science. *Commun. ACM* **37**, 3 (1994), 93-105.

Teini-ikäisten tyttöjen asenteet ICT-alaa kohtaan

Hanna Remula

Tiivistelmä.
Naispuolisten tietoteknisten alojen opiskelijoiden määrä 70-luvulla oli lähes puolet kaikista tietoteknisten alojen opiskelijoista. 80-luvulla määrä kääntyi laskuun, ja tämän jälkeen jälleen nousuun. Tällä hetkellä naisten osuus on edelleen suhteellisen pieni: noin neljäsosa ICT-alan opiskelijoista on naisia. Viime vuosina alan opiskelun suosio on naisten osalta jälleen kääntynyt laskuun. Naisten määrän vähäisyys kiinnostaa sekä tutkijoita, päättäjiä että yritysmaailman ihmisiä. Suomessakin on yritetty tehdä toimenpiteitä, jotta naisia saataisiin enemmän ICT-alalle. Ilmiön taustalla näyttäisi olevan tyttöjen asenteet, jotka muodostuvat ympäristötekijöiden vaikutuksesta ja toistavat itseään sukupolvelta toiselle. Tässä tutkielmassa on tarkasteltu, minkälaisia tutkimuksia tyttöjen asenteista tietotekniikkaa ja ICT-alaa kohtaan on tehty ja minkälaisia tuloksia näistä on saatu.

Avainsanat ja -sanonnat: ICT-ala, sukupuoli, tytöt, asenteet
CR-luokat: K.7.1

1. Johdanto

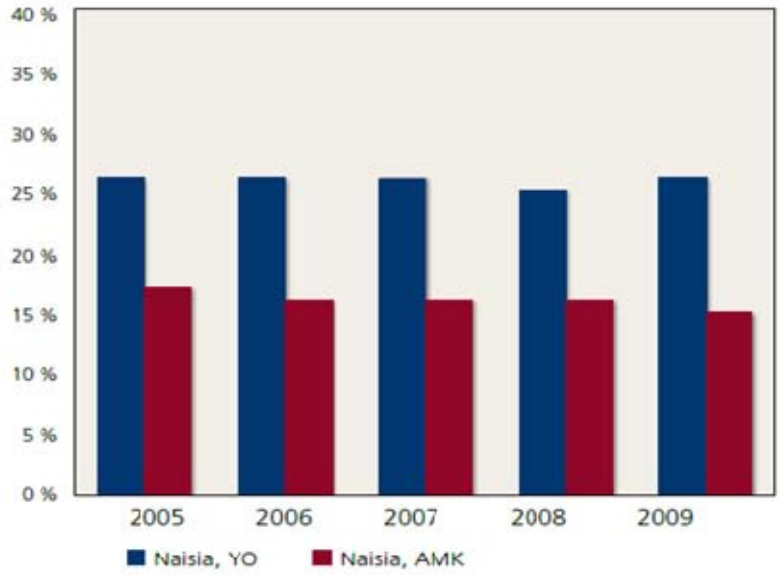
Naisten osuus tietoteknisten alojen opiskelijoista on pieni, mutta varsinkin ammattikorkeakoulutasolla tyttöjen määrä on jopa laskenut viime vuosina, kuten kuvasta 1 voidaan havaita. Kuitenkin tytöt ovatkin yhä enemmän kiinnostuneita tietotekniikan käytöstä varsinkin sosiaalisen kanssakäymisen muodoissa. ICT-alaa on perinteisesti pidetty maskuliinisena alana, vaikka naisia tietoteknisille aloille halutaankin.

ICT-alan täydellinen määrittely on hankalaa, sillä ala on suhteellisen uusi, ja siitä on historian aikana käytetty useita eri käsitteitä. Tässä tutkielmassa käytän termiä ICT-ala (*information and communications technology*) tai sen suomenkielistä vastinetta *tieto- ja viestintäteknologia*, joita nykyisin alasta useimmiten käytetään. Tämän termin lisäksi alaan ja sen opiskeluun viitataan myös termeillä IT-ala (*information technology*), IS (*information systems*) sekä CS (*computer science*). ICT-sektoriin kuuluvat alat määrittelen OECD:n suosituksen mukaan, joka perustuu YK:n toimialaluokituksen ISIC Rev.4:n toimialoihin [Tilastokeskus, 2012].

Tutkimus on rajattu koskemaan opiskelevia nuoria, jotka ovat tekemässä päätöksiä tulevaisuuden jatko-opiskeluidensa suhteen. Teini-ikäisiin, 13–19-vuotiaisiin, päädyttiin sen vuoksi, että tässä iässä nuoret alkavat tehdä päätök-

siä jatko-opiskelualastaan. Eri maiden koulujärjestelmissä on eroja, ja joissakin maissa päätöksiä tulevaisuuden uravalinnoista tehdään aikaisemmin kuin toisissa. Esimerkiksi Suomessa, vaikka lukion ja ammattikoulun loppupuolella tehdäänkin valinnat jatko-opiskeluiden suhteen, on silti jo aikaisemmin tehty valintoja, jotka suuntaavat opiskelua. Esimerkiksi lukion pitkän matematiikan taitojen hallintaa vaaditaan joissakin yliopistotasoisessa tekniikan opiskelussa jossain määrin, ja lyhyen matematiikan valitsemisen on kuviteltu rajaavan nämä vaihtoehdot kokonaan pois. Kuitenkin kaikissa tämän tutkielman sisältämien tutkimusten koulujärjestelmissä koulutusvalinnat tehdään 13–19-vuotiaana.

Eri maiden koulujärjestelmät aiheuttivat tutkimuksessa ongelmia myös nimityksien suomentamisten kohdalla. Olen pyrkinyt asettamaan kouluasteiden suomenkieliset nimet vastaamaan parhaimmalla mahdollisella keinolla suomalaista koulujärjestelmää ja tarkentamaan näitä tarvittaessa. Tarvittaessa olen myös käyttänyt ikähaarukointia ja alkuperäistä englanniksi käännettyä nimitystä hahmottamaan koulutusasteen luonnetta tarkemmin.



Kuva 1. Naisten hakeutuminen ICT-alan koulutukseen Suomessa [Teknologiateollisuus, 2011]

Tutkimustyön tarkoituksena on löytää syitä sille, miksi tytöt eivät hakeudu ICT-alalle. Teknologiateollisuuden selvityksestä [2011] selvisi, että suurin osa syistä liittyy tyttöjen asenteisiin ja käsityksiin IT-alasta. Tämän vuoksi tutkimuskysymykseni koskeekin nimenomaan teini-ikäisten tyttöjen asenteita ICT-alaa kohtaan: Onko tytöillä negatiivinen kuva ICT-alasta? Jos, niin mistä nämä

negatiiviset asenteet syntyvät? Mitkä tekijät vaikuttavat siihen, että ICT-alaa ei valita omaksi tulevaisuuden alaksi?

Tutkielman aluksi selostetaan tutkimusmenetelmää ja -prosessia, aloittaen artikkelien hausta ja valinnasta ja päättyen artikkelien analyysiin. Seuraavaksi kerrotaan taustaa ja metodologiaa jokaisesta tutkimuksessa tarkemmin käsiteltävästä artikkelista. Tämän jälkeen käsitellään tutkimusartikkelit siten, että näiden tulokset on järjestelty tutkimusartikkeleista löytyneiden yhteisten teemojen mukaan. Viimeisenä on pohdinta tutkimuksista sekä yhteenveto.

2. Menetelmä

2.1. Artikkelien valinta

Tutkimusmenetelmänä käytetään kirjallisuuskartoitusta. Tutkimustyö aloitettiin kartoittamalla tutkimusta kuvaavat asiasanat, joita käytettiin hakusanoina etsittäessä aiheesta aiemmin valmistuneita tutkimuksia.

Hakuprosessissa käytiin läpi seuraavat tietokannat: ACM Digital Library, Springerin elektroniset lehdet, ScienceDirect sekä IEEE Xplore -artikkeliviitekannat. Hakusanat kohdistettiin hakukoneesta riippuen tiivistelmään, otsikkoon ja asiasanoihin. Tarkemmin hakusanat ja näiden tulokset on listattu taulukkoon 1.

Mistä etsittiin?	Hakusanat	Rajaukset	Tuloksia
ACM Digital Library	girls + attitudes + it	-	935
	girls + attitudes + it	Abstract	15
	gender + attitudes + it + field	Abstract	83
	gender + attitudes + "it field"	Abstract	1
	girls + attitudes + information + technology + studies	Abstract	4
SpringerLink	"school girls" + attitudes + "information tecnology"	Abstract & Title	0
	"school girls" + attitudes + computers	Abstract & Title	0
	girls + attitudes + computers	Abstract & Title	15
	girls + perception* + "it field"	Abstract & Title	0
	girls + perception* + computers	Abstract & Title	4
	girls + perceptions + "it field"	Abstract & Title	0
Science Direct	school + girls + attitudes + information + technology	Abstract & Title & Keyw ords	4
	girls + attitudes + computers	Abstract & Title & Keyw ords	29
	students + attitudes + ict + gender	Abstract & Title & Keyw ords	9
	students + perceptions + ict + gender	Abstract & Title & Keyw ords	4
	gender + attitudes	A&T&KW in Computers&Education	43
	girls + attitudes	A&T&KW in Computers&Education	16
	girls + perception	A&T&KW in Computers&Education	9
	gender + perception	A&T&KW in Computers&Education	25
IEEE Xplore	school girls attitudes information technology	Abstract	898
	"school girls" + attitudes + "information technology"	Abstract	0
	girls + attitudes + "information technology"	Abstract	0
	gender + attitudes + school + computer + science	Abstract	6

Taulukko 1. Tietokannoista tehdyt haut ja niiden tulokset.

Hakukoneiden lisäksi hyödyllisiä tutkimuksia löytyi tutkimuksien viiteluetteloista. Lisäksi jotkin hakukoneet suosittelivat samankaltaisia artikkeleita muista tutkimuksista toista artikkelia lukiessa.

Hakutuloksista nousi usein esille Computers & Education -lehti. Tästä lehdestä sai tehtyä tuloksellisia hakuja myös käyttämällä laajempia hakusanoja, jolloin käyttökelpoisia artikkeleita löytyi pelkästään otsikoista päättelemällä.

Kun hakutulokset oli rajattu sanavalinnoilla mahdollisimman hyvin, artikkeleita rajautui pois ensin otsikon, tämän jälkeen tiivistelmän, ja lopulta itse artikkelin sisällön perusteella. Artikkeleista karsiutuivat pois ne, jotka eivät vastanneet omaa tutkimuskysymystäni tyttöjen asenteista tietotekniikkaa tai IT-alaa kohtaan, sekä ne, joissa tutkimuskohteena eivät olleet kouluikäiset tytöt. Näin jäljelle jäivät ne artikkelit, joiden sisältöön tässä tutkielmassa perehdytään tarkemmin.

2.2. Artikkelien analyysi

Artikkelit analysoitiin niiden koko tekstin perusteella. Jo aikaisemmin artikkeleista oli karsittu pois ne, joiden tutkimuskysymys ei vastannut omaani. Artikkelin analyysivaiheessa keskityttiin nimenomaan asenteisiin sukupuolta ja ICT-alaa kohtaan, ja mikäli tutkimuksessa oli myös jokin muu aihe, jätettiin se analyysin ulkopuolella.

Artikkelien analyysivaiheessa myös huomattiin, että huolimatta samantyylisestä tutkimuskysymyksestä oli asetelmissa eroja eri tutkimusten välillä. Tämä johtui usein siitä, että maiden koulujärjestelmät olivat erilaisia, ja näin ollen myös tietotekniikka oppiaineena esiintyi hyvin eri tavoin eri kouluissa. Tämän vuoksi päädyttiin selostamaan omassa luvun alikohdassaan taustoja jokaiselle tutkimukselle erikseen. Luvussa neljä artikkeleita on käsitelty yhdessä löytyneiden yhteisten teemojen mukaan jaoteltuna.

3. Tutkimukset

Tässä luvussa esitellään tutkimukset, joista kirjallisuuskatsaus muodostuu. Tutkimukset on esitelty tutkimusmaiden mukaisessa järjestyksessä, aloittaen Suomesta. Tämän jälkeen käymme läpi eurooppalaiset valtiot ja viimeiseksi Australian. Artikkeleissa on käytetty erilaisia termejä, kuten IT, IS, ICT, CS (viitaten lukuun 1). Olen pyrkinyt suomentamaan ja yhdenmukaistamaan näitä mahdollisimman tarkasti siten, että ne vastaisivat alkuperäistä merkitystään.

3.1. Suomi

3.1.1. Lukioikäisten tyttöjen asenteet IT-alaa kohtaan

Vaikka tytöt eivät ole kiinnostuneita tietojärjestelmien, ohjelmistotuotannon ja tietojenkäsittelyn opiskelusta, on olemassa vain muutamia pohjoismaisia tutkimuksia tämän ilmiön syistä. Leiviskä ja Siponen [2010] ovat Suomessa tutkineet lukioikäisten tyttöjen asenteita IT-alaa kohtaan. Tutkimusotos oli yhteensä 64 lukioikäistä eli 16–19-vuotiasta tyttöä. Aineisto kerättiin vuosina 2000 ja 2003 ja se koostui yliopiston järjestämän kirjoituskilpailun esseistä.

Tutkimuksessa käytettiin kvalitatiivista analyysia, mikä perusteltiin muun muassa sillä, että haluttiin tietää, mitä mieltä tytöt ovat oikeasti tieto- ja viestintäteknologiasta. Tällä tavalla haluttiin varmistaa, ettei tutkimuksessa itsessään rakenneta minkäänlaista teoreettista päämäärää, esimerkiksi kyselylomakkeen avulla, vaan annettiin tyttöjen itsensä kertoa omista asenteistaan.

3.1.2. Miksi tytöt eivät hakeudu IT-alalle: Teknologiateollisuuden tutkimushanke

Teknologiateollisuuden [2011] tutkimushanke *Naiset IT-alalla* sisältää tutkimuksen, joka selvittää tyttöjen näkemyksiä ICT-alasta sekä koulujen vaikutusta näihin. Tutkimuksessa käsitellään myös opettajien näkemyksiä. Tutkimusaineisto kerättiin keväällä 2010 kolmesta alakoulusta, kolmesta yläkoulusta ja kahdesta lukiosta Suomessa. Otos tutkimuksessa oli 149 tyttöä sekä 28 opettajaa.

Tutkimusmenetelmänä käytettiin monitapaustutkimusta, jolla pyrittiin saamaan mahdollisimman tarkkoja ja monipuolisia tuloksia. Menetelminä käytettiin tytöille kyselylomaketta sekä ryhmähaastattelua ja opettajille teemahaastatteluja. Kysymykset koskivat ICT-alan tuntemusta, mielikuvia ja kiinnostusta ICT-alaan, koulujen tieto- ja viestintäteknologian käyttöä sekä sitä, miten erot näkyvät niiden koulujen välillä, joissa tieto- ja viestintäteknologiaa käytetään enemmän opetuksessa. Kyselylomakkeen vastaukset analysoitiin siten, että kunkin koulun vastaukset yhdistettiin. Haastatteluista saatu aineisto jaoteltiin teemoittain vastaamaan tutkimuskysymyksiä.

Tässä tutkielmassa painopiste tämän tutkimushankkeen osalta on yläkoululaisten ja lukiolaisten osuus tutkimuksesta.

3.2. Kreikka

3.2.1. Poikien ja tyttöjen ICT-käsitykset: Onko opettajilla merkitystä?

Vekirin [2009] tutkimus *Boys' and girls' ICT beliefs: Do teachers matter?* käsittelee tyttöjen ja poikien käsityksiä tietotekniikasta, ja sitä, minkälaisia vaikutuksia opettajilla on näiden käsityksien muodostumisessa. Tutkimus on toteutettu

kreikkalaisella yläasteella (Gymnasium) vuosina 2007–2008. Tutkimusotoksena oli 301 kahdeksannen ja yhdeksännen luokan oppilasta (135 poikaa, 166 tyttöä), ja heidän seitsemän opettajaansa (neljä naista, kolme miestä) neljästä eri koulusta. Kyselylomaketutkimus sisälsi 26 Likert-asteikollista kysymystä, jotka käsittelivät oppilaiden omia käsityksiä tietotekniikkataidoistaan, heidän arvojaan ja uskomuksiaan sekä vanhemmilta saatua tukea. Lisäksi tutkittiin niitä odotuksia, joita oppilaat asettivat opettajaa kohtaan tietotekniikkaan liittyen.

Kreikassa tietotekniikka (IS, information science) on erillinen oppiaineensa, joka aloitetaan 7. luokalla, joka on kreikkalaisen yläasteen ensimmäinen luokka. Ensimmäiset kaksi vuotta opetus käsittelee perusasioita, kuten sähköpostin tai tekstinkäsittelyn käyttämistä, kun taas 9. luokalla perehdytään ohjelmointiin Logo-kielen avulla. Tietotekniikan opetussuunnitelman mukaan tarkoituksena on opettaa työelämässä tarvittavia taitoja.

3.2.2 Ovatko tietojenkäsittelytiede ja informaatioteknologia edelleen maskuliinisia aloja? Lukio-opiskelijoiden käsityksiä ja uravalintoja

Papastergioun [2007] tutkimus *'Are computer science and information technology still masculine fields?' High school students perceptions and career choices* käsittelee kreikkalaisten lukio-opiskelijoiden niitä aikeita ja motivaatiota, mitä tulee tietojenkäsittelytieteiden akateemisia opintoja kohtaan. Tutkimuksessa tarkkaillaan myös perheen ja kouluympäristön vaikutuksia ja oppilaiden käsityksiä tietojenkäsittelystä sekä informaatioteknologiasta ja pyritään etsimään sukupuolten välisiä eroja näiltä aloilta.

Tutkimusotos valittiin satunnaisesti viidestä kreikkalaisesta julkisesta lukiosta, ja otoksena oli 358 oppilasta (177 poikaa ja 181 tyttöä), jotka olivat iältään 17–18-vuotiaita. Opiskelijat olivat siis lukion viimeisellä luokalla ennen valmistumistaan toisen asteen koulutuksesta. He olivat jo valinneet suuntauksen opintoihinsa lukiossa, mutta eivät olleet vielä valinneet tarkkaa jatko-opiskeluainetta. Tutkimusaineisto kerättiin kyselytutkimuksen avulla. Tutkimuksessa kysymykset oli jaettu kuuteen ryhmään, ja niillä pyrittiin saamaan aikaan sekä kvalitatiivista että kvantitatiivista dataa.

3.3. Espanja

3.3.1. Sukupuolierot tietokoneasenteissa ja teknologiaan liittyvien ammatti-en valinnassa

Espanjassa 54 % yliopisto-opiskelijoista lukuvuotena 2008–2009 oli naisia, mutta naisten osuus oli vain 17 % tietojenkäsittelytieteen ja 26 % tietoliikennetekniikan opiskelijoista. Tutkimuksessa *Gender differences in computer attitudes and the choice of technology-related occupations in a sample of secondary students in*

Spain [Sáinz and López-Sáez, 2009] on tutkittu espanjalaisten koululaisten sukupuolieroja tietokoneasenteissa ja teknologiaan liittyvien ammattien valinnassa.

Tutkimusotoksena oli 550 12–20-vuotiasta (keski-ikä 15 vuotta) espanjalaisen yläkoulun ja lukion opiskelijaa, joista 252 oli poikia ja 298 tyttöjä. Tutkimukseen osallistuneille toimitettiin kyselytutkimus, joka sisälsi sekä sosiodemografisia että asenteellisia muuttujia. Tarkoituksena oli selvittää eri tekijöiden merkitykset oppilaiden asenteisiin ja valintoihin teknologiaan liittyen.

3.3.2. Vanhempien ja opettajien käsitykset ICT-ammateista, sukupuolieroista ja roolistaan opiskeluvalinnoissa

Opettajien ja vanhempien vaikutusta jatko-opiskeluvalintoihin ei ole juuri Espanjassa dokumentoitu. Sáinz ja muut [2011] ovatkin siis analysoineet, kuinka tärkeäksi opettajat ja vanhemmat itse kokevat roolinsa tässä prosessissa. Sáinz ja muut olivat myös kiinnostuneita siitä, minkälainen näkemys vanhemmilla ja opettajilla on ICT-ammateista.

Tutkimusotoksena oli seitsemän kohderyhmää, joissa neljässä oli 27 vanhempaa ja kolmessa 22 toisen asteen opettajaa (12–16-vuotiaiden). Nämä kohderyhmät muodostettiin viidestä koulusta, jotka sijaitsivat Katalonian kaupunki- ja maaseutualueilla.

Tutkimukseen osallistujia pyydettiin osallistumaan koulutapaamiseen, jossa keskusteltiin nuorten tulevaisuuden koulutusvalinnoista. Kutsu lähetettiin kaikille yläkoulun (junior secondary school) viimeisellä luokalla opiskelevien nuorten vanhemmille. Kohderyhmille esitettiin seuraavat kysymykset, joihin vastattiin kirjallisesti:

Mitä tulee mieleen, kun ajattelet henkilöä, joka työskentelee ICT-alalla? Valitsevatko naiset ja miehet erilaisia opiskeluvaihtoehtoja? Jos, niin miksi? Mitkä uskotte olevan syyt siihen, että tytöt valitsevat enemmän humanistisia aloja ja pojat teknologia-aloja? Mitä ajattelette siitä, että vain 17 % ICT-alan opiskelijoista on tyttöjä? Millaiseksi näet oman roolisi lapsien tai opiskelijoiden akateemisissa valinnoissa? Mitkä muut tekijät vaikuttavat näihin valintoihin?

3.4. Alankomaat

3.4.1. Sukupuolierot tietokoneasenteissa: Onko koululla merkitystä?

Alankomaissa vain noin 10 % ICT-alalla työskentelevistä on naisia. Meelissen ja Drent [2007] ovat analysoineet tutkimusaineiston vuoden 1999 ICT-seurannassa. ICT-seuranta on hollantilainen laajamittainen tutkimus tieto- ja viestintätekniikan käytöstä kaikilla kouluasteilla. ICT-seuranta aloitettiin vuonna 1997, ja se

on toistettu vuosittain, jotta voitaisiin arvioida koulutuspolitiikan vaikutuksia tietotekniikan käyttöön koulutuksessa.

Meelissen ja Drent olivat kiinnostuneita nimenomaan siitä, minkälainen vaikutus koululla on sukupuolten välisiin eroihin tietokoneisiin liittyvissä asenteissa. Yleensä esimerkiksi naisopettajan varmat ICT-taidot vaikuttavat tyttöjen käsityksiin positiivisesti. ICT-seurannassa kerätään tietoa tieto- ja viestintätekniikan käytöstä koulutuksellisiin tarkoituksiin, sekä myös taustatietoa koulujen, opettajien ja oppilaiden kyselytutkimusten avulla. Koulut valitaan satunnaisotannalla. Tutkimusotos vuonna 1999 koostuu 209 koulusta ja 3 893 oppilaasta (1 987 tyttöä ja 1 906 poikaa).

3.5. Saksa

3.5.1. Kuinka sukupuoli vaikuttaa näkemykseen (omista) tietokonetaidoista?

Sieverkind ja Koch [2008] ovat tutkineet, arvioidaanko naisten tietotekniset taidot vähemmän myönteiseksi kuin miesten. Psykologisen tutkimuksen tarkoituksena oli selvittää, vaikuttaako sukupuoli siihen, miten tietokonetaitoja arvioidaan; sekä muiden arvioidessa toisten kykyjä kuin itsearvioinninkin yhteydessä. Tutkimusotos käsitti Berliinin avoimessa yliopistossa 206 opiskelijaa, joista 99 miehiä ja 107 naisia. Psykologian opiskelijoita oli 113 ja 93 muiden alojen edustajia. Tutkimukseen osallistuneiden opiskelijoiden ikien keskiarvo oli 23,7 vuotta (keskihajonta = 4,2).

Tutkimukseen osallistujat jaettiin 4-8 hengen ryhmiin. Osallistujat näkivät videon, jossa joko mies tai nainen suoritti tietokoneella saman tehtävän siinä onnistuen. Video pysäytettiin sen alkuvaiheessa, jolloin osallistujilta kysyttiin ennuste siitä, kuinka hyvin videolla esiintyvä henkilö tämän tehtävän suorittaisi. Videon lopuksi he ensin arvioivat esiintyneen henkilön suorituksen, ja tämän jälkeen vertasivat omaa hypoteettista suoriutumistaan samasta tehtävästä videolla esiintyneen henkilön suoritukseen nähden.

3.6. Australia

3.6.1. Miksi lukiotytöt välttävät ammatillisesti suuntautuneita IT-aineita?

Myös Australiassa naiset valitsevat kansainvälisen trendin mukaan vähemmän IT-ammatteja ja -suuntauksia opinnoilleen. Anderson ja muut [2006] ovat tutkimuksessaan *'Because it's boring, irrelevant and I don't like computers': Why high school girls avoid professionally-oriented ICT subjects* selvittäneet syitä sille, minkä vuoksi tytöt eivät valitse tietoteknisiä aineita lukiossa ja suuntaudu IT-alalle.

Tutkimuksessa on keskitytty kahteen viimeiseen kouluvuoteen, jolloin oppilaat tekevät ainevalintoja, jotka niin edelleen johtavat urasuuntauksiin jatko-opiskelun suhteen. Tutkimus käsittelee nimenomaan tietotekniikan edistyneitä kursseja, jotka johtavat IT-alan urapolkuun. Ammatillisilla ICT-urilla on tutkimuksessa viitattu seuraaviin tehtäviin: ohjelmisto- ja laitteistotuotanto, tietokoneiden tekninen tuki sekä verkkojärjestelmien ja tietokantojen luominen ja hallinnointi. [Sandy and Burger, 1999].

Tutkimus on toteutettu vuonna 2005, ja sen otos käsitti 1 453 tyttöä 12. ja 13. luokka-asteelta Queenslandista, Australiasta. Otos jaoteltiin IT-aineiden valitsijoihin (131 tyttöä) ja ei-valitsijoihin (1 322 tyttöä). IT-aineiden valitsijoille esitettiin kyselylomakkeessa positiivisia väittämiä ja ei-valitsijoille negatiivisia väittämiä, joihin annettiin vastaukset viisiportaisella Likert-asteikolla. Näitä vastauksia vertailtiin toisiinsa.

3.6.2. Oppilaiden ICT-käsitykset: Analyysi sukupuolten välisistä eroista

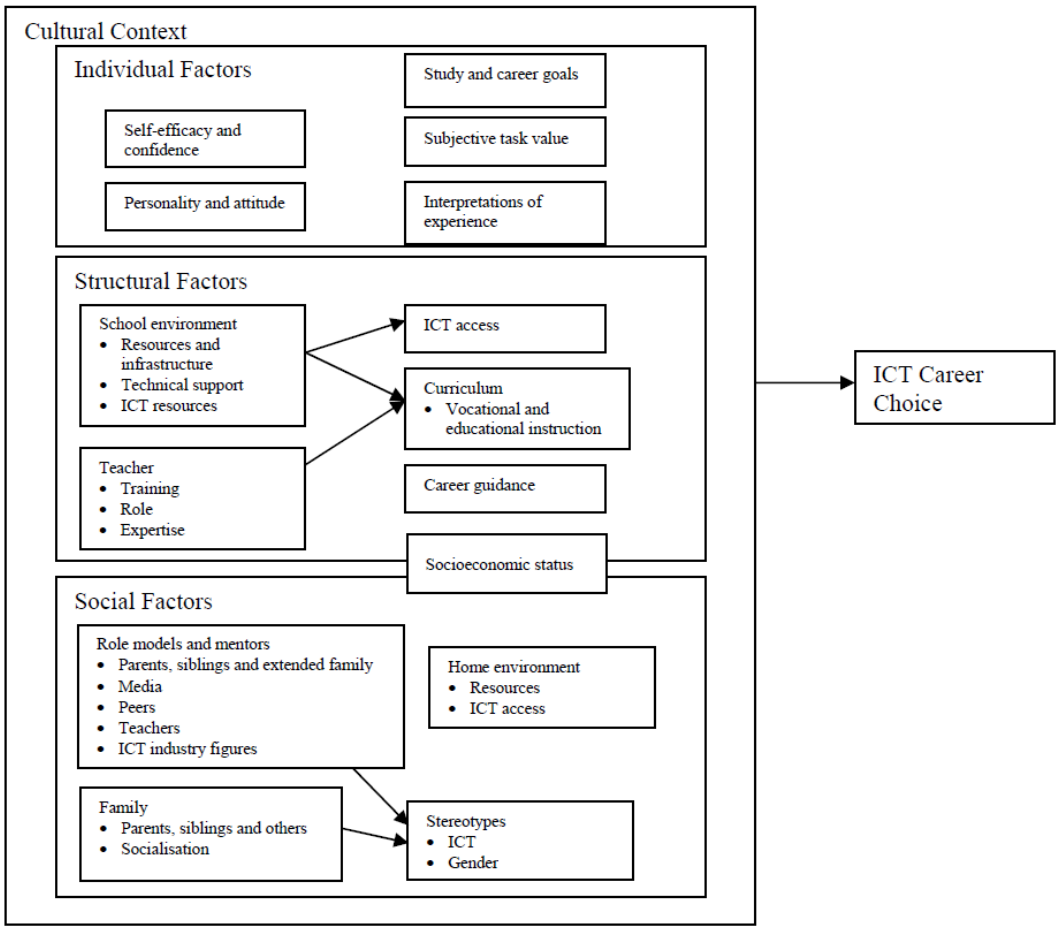
McLahlan ja muut [2010] ovat tutkineet toisen asteen opiskelijoiden asenteita ja odotuksia tietotekniikkaa kohtaan. Tutkimusotoksena oli 681 yliopiston ensimmäisen vuoden opiskelijaa, joista 506 eivät olleet valinneet tieto- ja viestintätekniikkaa toisen asteen koulussa oppilaitoksessa, ja jäljellejäävät 175 olivat valinneet edellä mainittuja aineita aikaisemmassa pakollisessa oppilaitoksessa. Vaikka aineisto kerättiin yliopisto-opiskelijoista, sitä kuitenkin verrattiin heidän valintoihinsa toisen asteen koulutuksessa.

Anonyymi kyselylomake toimitettiin ja kerättiin yliopistolla neljän viikon aikana vuonna 2008. Kysymyksiin vastattiin joko viisiportaisen Likert-asteikon avulla, kategorian valinnalla tai vastaamalla kysymykseen kirjallisesti.

3.6.3. Sukupuolistereotypiat vallitsevat ICT-alalla: Tutkimustarkastelu

Clayton ja muut [2009] ovat tutkineet sukupuolen stereotypisointia ja tietotekniikkaa, sekä sitä kuinka se vaikuttaa tyttöjen osallistumiseen ICT-alan koulutukseen ja työhön. Tutkimusmetodina on käytetty kirjallisuustarkastelua.

Vaikka moni asia vaikuttaa aine- ja uravalintoihin (vrt. kuva 1), tässä tutkimuksessa on perehdytty nimenomaan sukupuolen stereotypisointiin ja median tärkeään rooliin siinä. Pääpaino tässä tutkimuksessa on australialainen tutkimuskirjallisuus koskien kouluikäisiä lapsia ja heidän aine- sekä uravalintojaan. Lisäksi on otettu mukaan jonkin verran myös kansainvälisiä tutkimuksia, joissa kulttuurinen konteksti vastaa Australian olosuhteita.



Kuva 2. Tekijät tyttöjen aine- ja uravalintojen takana. [Clayton et al. 2009; alkuperäinen lähde: Clayton, 2007]

4. Teemat

Edellisessä luvussa käsiteltiin kirjallisuuskartoituksessa mukana olevat tutkimukset. Tässä luvussa esitetään näiden tutkimusten tulokset eriteltynä neljään eri aihepiiriin: tyttöjen asenteet tietotekniikan opiskelua kohtaan, tyttöjen asenteet ICT-alaa ja -ammattilaisia kohtaan, vanhempien ja opettajien suhtautuminen ICT-alaa kohtaan, tyttöjen suhtautuminen tietokoneita kohtaan sekä tyttöjen näkemykset omista tieto- ja viestintätekniikkataidoistaan. Tutkimusten tulokset on myös taulukoitu tutkimuskohtaisesti taulukkoon 2.

		Tietotekniikan opiskeluun liittyvät asenteet	IT-ala- / -ammattilaisasenteet	Koulun, opettajien ja vanhempien vaikutus	Tietotekniikkataidot
Suomi	*Teknologiateollisuus, 2011*	- Ei tietoa alan opiskelusta	- 'Nörttejä', ei tietoa alasta	- Naisroolimallien puute - Tiedon vähäisyys kouluissa	- Peruskäyttötaidot hataria
	Leiviskä and Siponen, 2010		- Kehittyvä, hyvin palkattu - Antisosiaalisuus, ei ihmisläheisyyttä		
Kreikka	*Vekiri, 2009*			- (Nais)Opettajilta ja vanhemmilta saatu kannustus lisää positiivisia asenteita	
	Papastergiou, 2007		- Taattu työllistyminen - Ei ihmisläheisyyttä - Vaikeahko ala	- Miespuolinen perheenjäsen usein tietotekniikan käyttäjänä	- Omat ajatukset tietotekniikan käyttötaidoista vaikuttavat alalle
Espanja	*Sáinz and López-Sáez, 2009*		- Pojat pitävät älykkäämpinä - Tytöt sosiaalisempina		
	Sáinz et al., 2011			- Vanhemmat ja opettajat: stereotyyppinen näkemys - Ajattelevat suurimman syyn löytyvän mediasta - Oman roolin kuvitellaan olevan vähäinen	
Alankomaat	*Meelissen and Drent, 2007*			- (Nais-)opettajilta ja anhemmilta saatu kannustus lisää positiivisia asenteita	- Pojilla stereotyyppisempi näkemys tietokoneen käyttötaidoista
Saksa	*Sieverkind and Koch, 2008*				- Itsearvioinnissa miehet arvioivat itsensä paremmiksi kuin naiset itsensä
Australia	*Anderson et al., 2006*	- Tylsä - Ei hyötyä työuralle			
	McLahlan et al., 2010	- Epäkiinnostus - Tylsyys	- Ei kiinnostusta alaa kohtaan	- Tytöillä vanhempien ja opettajien rooli ammatinvalinnassa on suurempi	- Miehet kokivat parempaa tuntemusta ja luottamusta tietokoneita kohtaan
	Clayton et al., 2009			- Naisroolimallien puuttuminen lisää stereotyyppistä näkemystä	- Stereotypisointi vaikuttaa tietotekniikkataitojen itsearviointiin

Taulukko 2. Tutkimusartikkelien teemat ja näiden tulokset.

4.1. Tietokoneisiin ja tietotekniikkataitoihin liittyvät asenteet

Tietokoneet eivät kiinnosta tyttöjä [Andersson et al., 2006]. Tytöillä on silti positiivinen kuva tieto- ja viestintätekniikasta, mutta ei niin positiivinen kuin pojilla [Sáinz and López-Sáez, 2009; Meelissen and Drent, 2009]. Joidenkin tutkimusten mukaan tyttöjen ja poikien tietokoneen käytössä ei ole eroja [McLahlan et al., 2010], ja joidenkin mukaan pojat käyttävät tietotekniikkaa enemmän kuin tytöt [Sáinz and López-Sáez, 2009; Papastergiou, 2007].

Sieverkindin ja Kochin [2008] mukaan sukupuolella ei ole väliä, kun arvioidaan toisten tietoteknisiä taitoja. Meelissenin ja Drentin [2009] mukaan taas pojat ajattelevat tietotekniikkataidoista stereotyyppisemmin kuin tytöt: pojat pitävät poikia parempina tietotekniikan käyttäjinä. Sukupuolierot kuitenkin nousevat esiin lähes aina silloin, kun kyseessä on omien taitojen arviointi. Tytöt pitävät omia taitojaan huonompina kuin pojat omia taitojaan [McLahlan et al., 2010; Sieverkind and Koch, 2008; Papastergiou, 2007]. Papastergiou [2007] on havainnut myös yhteyden tietoteknisten taitojen itsearvioinnin ja alalle hakeutumisen välillä. Stereotypisoinnilla on nähty negatiivinen yhteys myös tietoteknisten taitojen itsearviointiin [Clayton et al., 2009].

Teknologiateollisuuden [2011] teettämän tutkimushankkeen mukaan, on opiskelijoiden tieto- ja viestintätekniikan perustaidoissaan kuitenkin huomattavia ongelmia, vaikka heitä kutsutaankin diginatiiveiksi.

4.2. Tietotekniikan opiskeluun liittyvät asenteet

Tässä kohdassa käsitellään tyttöjen asenteita ja mielipiteitä tietotekniikan opiskelua kohtaan. Tietotekniikan opiskelulla tässä luvussa tarkoitetaan sekä opiskelua perus- ja lukiotason oppilaitoksissa että jatko-opiskeluvalintana korkeakoulussa.

Korkeakoulutasolla länsimaissa naisten pieni osuus ICT-alan opiskelijoista on yleinen trendi. Osa tytöistä kertoi syyksi sen, että he eivät ole päässeet tutustumaan tietotekniikkaan kotona ja koulussa tarpeeksi [McLahlan et al., 2010].

Tytöt valitsevat tietotekniikkaa valinnaisena aineena vähemmän myös perus- ja lukioasteella. Valitsematta jättämisen syynä on ennemminkin se, että ainetta pidetään epäkiinnostavana, eikä niinkään vaikeana. [McLahlan et al., 2010]. Tyttöjen mielestä tietotekniikka on tylsää, eikä siitä kuvitella olevan hyötyä tulevaisuuden uraa ajatellen [Anderson et al., 2006].

4.3. ICT-alaan ja -ammattilaisiin liittyvät asenteet

Vaikka tytöt hakeutuvat ICT-alalle vähemmän kuin pojat, pitävät he silti alaa yhtä sopivina sekä naisille että miehille. Tytöillä ei kuitenkaan ole tarpeeksi tietoa siitä, mitä työtehtäviä ICT-ala pitää sisällään [Teknologiateollisuus, 2011]. Clayton ja muut [2009] ovatkin todenneet, että media paikkaa tätä epätietoisuutta negatiivisilla mielikuvilla, ja mielikuva alan ja sen ammattilaisten stereotyyppisestä luonteesta vähentää alalle hakeutumista. Papastergioun [2007] mukaan pojilla toisaalta on enemmän stereotyyppisiä näkemyksiä alasta: he esimerkiksi pitävät alaa sopivampana miehille sekä uskovat siihen, että miehet pärjäävät alalla paremmin kuin naiset. Stereotyyppiset mielikuvat ICT-ammattilaisista elävät myös tyttöjen keskuudessa edelleen: tytöt pitävät alan ammattilaisia epäsosiaalisina nörtteinä. Alan työtehtäviä pidetään myös edelleen epäso-

siaalisina – vuorovaikutussuhteet ovat lähinnä ihmisen ja koneen välisiä [Teknologiateollisuus, 2011; Leiviskä and Siponen, 2010]. Onkin useasti huomautettu, että tytöille tärkeät naisroolimallit puuttuvat alalta lähes täysin [Teknologiateollisuus, 2011; Leiviskä and Siponen, 2010; Sáinz et al., 2011]. Clayton ja muut [2009] ovatkin huomanneet, että naisroolimallien puuttuminen lisää jo entisestään vallalla olevia stereotyyppisiä näkemyksiä ICT-alasta. Alaa pidetään myös tylsänä ja epäkiinnostavana, mikä aiheuttaa sen, ettei alan jatkokoulutukseen hakeuduta [McLahlan et al., 2010].

Toisaalta ICT-alaa pidetään myös hyväpalkkaisena, korkeasti arvostettuna, kasvavana ja kehittyvänä, sekä sen työtilannetta pidetään yleisesti hyvänä. ICT-ammattilaisia tytöt pitävät myös taitavina, osaavina ja älykkäinä sekä korkeasti koulutettuina [Leiviskä and Siponen, 2010]. Alan turvattu työllisyystilanne on monelle tytölle tärkein syy hakeutua alan koulutukseen [Papastergiou, 2007].

4.4. Vanhempien ja opettajien asenteet ICT-alaa kohtaan

Vanhempien käsityksillä ja uskomuksilla on suuri vaikutus siihen, miten oppilaiden minäkuva opiskelun suhteen kehittyy. Näissä käsityksissä ja uskomuksissa on myös usein sukupuolten välisiä eroja. [Parsons et al., 1982]. Varsinkin, kun mediasta ICT-alan naisroolimallit puuttuvat, opettajilla ja vanhemmilla on yhä suurempi rooli. Tytöt pitävät opettajia ja vanhempia tärkeinä mielipidevaikuttajina tulevaisuuden alavalintoja kohtaan, kun taas pojille kavereiden vaikutus on suurempi [McLahlan et al., 2010]. Lapset pitävät sekä vanhempiaan että opettajiaan roolimalleina, jolloin on tärkeää myös tietää, millä tavoin heidän näkemyksensä ICT-alasta heijastuvat tyttöjen tulevaisuuden valintoihin. Huomattavaa on, etteivät opettajat ja vanhemmat kuitenkaan välttämättä itse pidä omaa rooliaan lasten uravalinnoissa erityisen suurena [Sáinz et al., 2011].

Myös vanhemmilla on ICT-alasta stereotyyppisiä näkemyksiä. Varsinkaan äidit eivät pidä alaa viehättävänä, ja alan epäsosiaalisia ominaisuuksia ei koeta yhteensopiviksi naissukupuolelle tyypillisten ominaisuuksien kanssa. Äitien mielestä alan miesvaltaisuus sekä tyttöjen huonommat matemaattiset taidot vaikeuttavat alalle hakeutumista. Sekä vanhemmat että opettajat pitävät naisroolimallien puuttumista ja median luomaa stereotyyppistä naiskuvaa suurimpina syinä sille, etteivät tytöt hakeudu ICT-alalle [Sáinz et al., 2011]. Meelissen ja Drent [2007] korostavat myös että vanhemmilta saatu kannustus lisää positiivisempia asenteita ICT-alaa kohtaan, mutta pojat kokevat saavansa kannustusta enemmän kuin tytöt.

Opettajilta ja vanhemmilta saatu kannustus korreloi tieto- ja viestintätekniikkataitojen itsearvioinnin kanssa lisäten positiivisempia mielikuvia omista taidoista. Naisopettajien asenteet vaikuttavat suuresti tyttöjen ICT-asenteisiin, kun taas miesopettajien asenteet vaikuttavat positiivisesti nimenomaan poikien

arviointiin omista tietotekniikkataidoistaan. [Vekiri, 2009]. Myös Meelissen ja Drent [2009] toteavat, että naisopettajien tietotekniikkataidoilla ja -asenteilla on positiivinen vaikutus tyttöjen mielikuviin.

Tyttöjen lisäksi myös opettajat kuvailevat, että heidän ja oppilaanohjauksen tiedoissa ICT-alasta on puutteita, ja tietoa pitäisi alasta saada esimerkiksi yritysvieraiden avulla [Teknologiateollisuus, 2011].

5. Pohdinta

Asenteet ICT-alaa kohtaan ovat yllättävän samankaltaiset länsimaissa, maasta tai tutkimustavasta riippumatta. Stereotyyppinen näkemys alasta on voimassa sekä oppilailla että vanhemmilla, mistä voikin päätellä stereotyyppisten näkemysten uusiutuvan sukupolvelta toiselle. Clayton ja muut [2009] kritisoivat myös voimakassanaisesti median tarvetta uudistaa ja vahvistaa stereotyyppistä näkemystä ICT-alasta samaan aikaan, kun naisten määrä alalla on pieni.

On myös huomattavaa, että ICT-alan määritelmä on edelleen epäselvä, eikä tytöillä ole selkeätä kuvaa alan tarjoamista mahdollisuuksista. Tällöin on tietenkin luonnollista, että mikäli tietotekniikka ei muutenkaan kiinnosta, alalle tuskin hakeudutaan, kun ei oikeastaan edes tiedetä, mitä se on.

Monissa tutkimuksissa myös korostui tyttöjen (ja vanhempien) näkemys ICT-alasta epäsosiaalisena ja pelkästään tietokoneen kanssa työskentelynä. Kuitenkin samaan aikaan vanhemmat ajattelevat naisten vähäisyyden ICT-alalla johtuvan lähinnä median luomasta stereotyyppisestä naiskuvasta. Tämä osoittaa sen, että asenteet ovat usein hyvin pinttyneitä, eikä niitä ole itsekään helppo havaita. Asenteet kuitenkin tunnetusti uusiutuvat helposti vanhemmilta lapsille, joten olisikin tärkeää lisätä ICT-alan tietoisuutta sekä kouluissa että myös mediassa.

Olisi myös hyvä korostaa myös nimenomaan käytettävyyspuolta ICT-aloilla, jolloin myös alan ihmisläheisyys tulisi paremmin tietoisuuteen. Vuorovaikutteisen ja ihmisläheisen teknologian opiskelun mahdollisuutta olisi tehtävä paremmin tunnetuksi varsinkin tytöille. Myös naisroolimallien puutetta harmiteltiin, joten sekä median että koulun sekä yritysten tulisi tuoda esille myös tytöille sopivia roolimalleja ICT-alalta. Tämän hetkinen mielikuva ICT-ammattilaisista nörtteinä kun ei houkuta tyttöjä alalle.

Suomessa on toteutettu vain vähän tieteellisiä tutkimuksia tyttöjen ICT-asenteista. Pro gradu -tutkielmassani tarkoitukseni on tutkia nimenomaan sitä, minkälaiset tekijät vaikuttavat tyttöjen luomaan kuvaan ICT-alasta. Olisi myös mielenkiintoista tutkia vanhempien ja opettajien näkemystä alasta, ja verrata tätä tyttöjen mielikuviin, ja näin selvittää, onko näillä yhteyksiä myös Suomessa.

6. Yhteenveto

Tutkimuksien tulokset vahvistavat Teknologiateollisuuden [2011] tuottamaa selvitystä, jonka mukaan tytöt pitävät IT-alaa tylsänä ja epämieluisena alana. Tyttöjen asenteet tietokoneita kohtaan ovat positiivisia, mutta IT-alaa ammattina ei haluta valita itselle. Edelleen on vahvasti voimassa stereotyyppinen näkemys IT-ammattilaisista: heitä pidetään epäsosiaalisina nörtteinä. Erilaiset roolimallit varsinkin naisille ja tytöille puuttuvat. Ylipäänsä käsitteet IT/ICT-ala tuntuvat olevan hämärän peitossa, ja näitä epäselvyyksiä paikataan negatiivisilla stereotypioilla. IT-alassa ei kuvitella olevan mitään sosiaalista tai ihmisläheistä, vaan kaikki työ tapahtuu tietokoneiden kanssa – ja ammatin ihmisläheisyys on nimenomaan tytöille tärkeä syy valita tulevaisuuden alansa.

Viiteluettelo

[Anderson et al., 2008] Neil Anderson, Colin Lankshear, Carolyn Timms and Lyn Courtney, 'Because it's boring, irrelevant and I don't like computers': Why high school girls avoid professionally-oriented ICT subjects. *Computers and Education* **50**, 5 (May 2008), 1304–1318.

[Clayton, 2007] K. Clayton, T*he influence of metropolitan Brisbane middle-school ICT experiences on girls' ICT study and career choices*. Doctoral Thesis. Griffith University, (2007).

[Clayton et al., 2009] Kaylene L. Clayton, Liisa A. von Hellens and Sue H. Nielsen, Gender Stereotypes prevail in ICT: a research review. In: *Proceedings of the 47th Annual Conference on Computer Personnel Research*, 153–158.

[Hou et al., 2006] Weimin Hou, Manpreet Kaur, Anita Komlodi, Wayne G. Lutters, Lee Boot, Sheila R. Cotten, Claudia Morrell, A. Ant Ozok and Zeynep Tufekci, "Girls don't waste time": pre-adolescent attitudes toward ICT. In: *Proceeding of Human Factors in Computing Systems CHI '06*, Extended Abstracts (2006), 875–880.

[Lasen, 2010] Michelle Lasen, Education and career pathways in information communication technology: What are schoolgirls saying?. *Computers & Education* **54**, 4 (May 2010), 1117–1126.

[Leiviskä and Siponen, 2010] Katja Leiviskä and Mikko Siponen, Attitudes of sixth form female students toward the IT field. *ACM SIGCAS Computers and Society* **40**, 1 (Mar. 2010), 34–49.

[Looker, 2008] E. Dianne Looker, Gender and information technology. *International Handbook of Information Technology in Primary and Secondary Education* **20**, 8 (2008), 779–788.

[McLachlan et al., 2010] Christine McLachlan, Annemieke Craig and Jo Coldwell, Student perceptions of ICT: a gendered analysis. In: *Proceedings of*

the Twelfth Australasian Conference on Computing Education (ACE ´10), 127–136.

[Meelissen and Drent, 2008] Martina R. M. Meelissen and Marjolein Drent, Gender differences in computer attitudes. *Computers in Human Behavior* **24**, 3 (May 2008), 969–985.

[Papastergiou, 2008] Marina Papastergiou, Are computer science and information technology still masculine fields? High school students' perceptions and career choices. *Computers & Education* **51**, 2 (Sep. 2008), 594-608.

[Parsons et al., 1982] Jacquelynne Eccles Parsons, Terry F. Adler and Caroline M. Kaczala, Socialization of Achievement Attitudes and Beliefs: Parental Influences. *Child Developments* **53**, 2, (Apr. 1982), 310-321.

[Sáinz and López-Sáez, 2010] Milargos Sáinz and Mercedes López-Sáez, Gender differences in computer attitudes and the choice of technology-related occupations in a sample of secondary students in Spain. *Computers & Ecucation* **54**, 2 (Feb. 2010), 578–587.

[Sáinz et al., 2012] Milargos Sáinz, Rachel Pálmen and Sara Carcía-Cuesta, Parental and secondary school teachers perceptions of ICT professionals, gender differences and their role in the choice of studies. *Sex Roles* **66**, 3-4 (Feb. 2012), 235–249.

[Sandy and Burger, 1999] M. Sandy and C. Burger, Women and minorities in information technology forum: Causes and solutions for increasing the numbers in the information technology pipeline. *The National Science Foundation, "Transitions from Childhood to the Workforce" Forum*, (Nov. 2009)

[Sieverding and Koch, 2009] Monika Sieverding and Sabine C. Koch, (Self-) Evaluation of computer competence: How gender matters. Computers & Education 52, 3 (Apr. 2009), 696–701.

[Teknologiateollisuus, 2011] Teknologiateollisyys ry, Naisia ICT-alalle! -selvitys, 2011.

[Tilastokeskus, 2012] Tilastokeskus, ICT-sektori. (katsottu: 12.12.2012) http://www.stat.fi/meta/kas/ict_sektori.html

[Trauth et al., 2012] Eileen M. Trauth, Curtis C. Cain, K.D. Joshdi, Lynette Kvasny and Kayla Booth, Embracing intersectionality in gender and IT career choice research. In: *Proceedings of the 50th Annual Conference on Computers and People Research*, 199–212.

[Vekiri, 2010] Ioanna Vekiri, Boys' and girls' ICT beliefs: Do teachers matter. *Computers & Education* **55**, 1 (Aug. 2010), 16–23.

Go-peliä pelaavista ohjelmista

Ville Riikonen

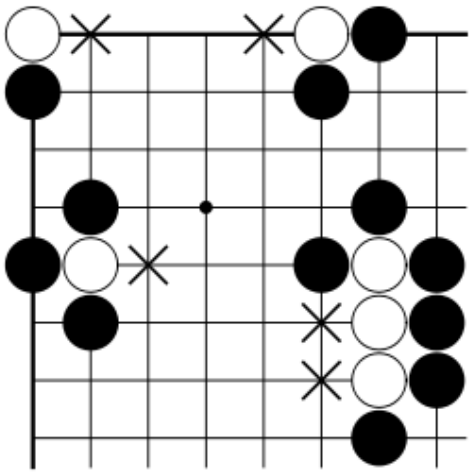
Tietokone-go on ollut yksi vaikeimmista haasteista tekoälyohjelmoinnin saralta. Go-lautapeliin ei voi soveltaa perinteisten lautapelien (kuten esim. shakin) tekoälystrategioita. Vaikka sinänsä go-pelin säännöt ovat hyvin yksinkertaiset, niin kokonaisen normaalin kokoisen pelilaudan eri pelitilanteiden määrä on valtava. Etenkin, kun pelin edetessä laudalta poistetaan ja lisätään nappuloita mielivaltainen määrä, on laskentakuormitus tietokone-ohjelmalle huomattava.

Avainsanat ja -sanonnat: Go, tietokone-go, lautapeli, tekoäly
CR-luokat: I.2.1, K.8.0

1. Mikä on go?

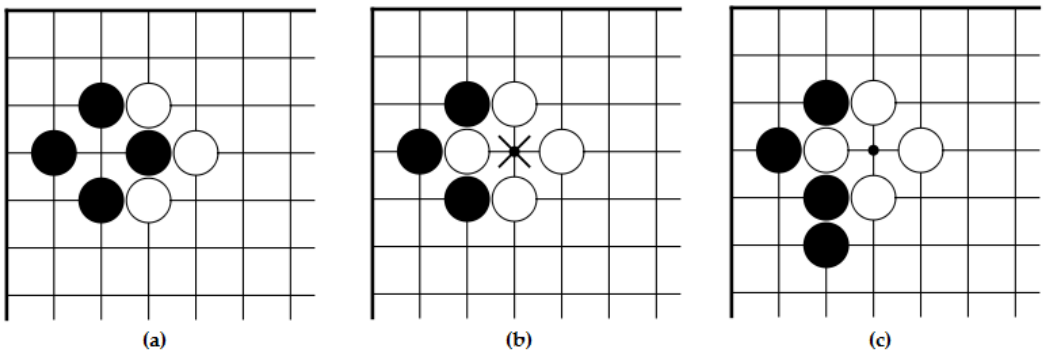
Go on alun perin kiinalainen lautapeli, jossa pelaajat yrittävät vallata nappuloillaan tyhjää aluetta pelilaudalta. Pelin arvellaan syntyneen n. 4000 vuotta sitten, mutta tästä ei ole luotettavaa kirjallista tietoa jäljellä [Macfadyen, 1998]. Gota pelataan laudalla, jossa on 19 x 19 suoran viivan muodostama ristikko. Myös 9 x 9 ja 13 x 13 viivan kokoiset laudat ovat tavanomaisia. Pelinappuloina toimivat mustat ja valkoiset linssin muotoiset nappulat, joita kutsutaan kiviksi. Kivet pelataan laudalla viivojen risteyskohtiin. Yleensä tasoltaan heikompi pelaaja pelaa mustilla kivillä ja vahvempi pelaaja pelaa valkoisilla kivillä. Pelaajat lisäävät vuorotellen oman värisiään kiviä laudalle. Kun kivi on pelattu, sitä ei voi enää liikuttaa. Jos kivi vangitaan, niin se poistetaan laudalta. Pelin alussa lauta on tyhjä, ja siirtojen määrää ei ole rajoitettu. Pelaajat valtaavat aluetta kivillään, ja peli loppuu, kun kumpikaan ei enää pelaa kiviä laudalle tai alueita ei voi enää vallata. Gossa ei ole lainkaan satunnaisia tapahtumia tai liikkeitä [Paatero, 2008]. Edellä mainitut säännöt mukaan lukien gossa on kaksi perussääntöä:

> i) Vapaussääntö: jokaisella kivellä pitää olla vähintään yksi vierekkäinen vapaus (piste laudalla) ylhäällä, alhaalla, vasemmalla tai oikealla, tai kiven pitää olla yhdistettynä kiviryhmään, jolla on vähintään yksi vierekkäinen vapaus. Kiviryhmät ja kivet poistetaan laudalta, jos niillä ei ole yhtään vapauksia. Havainnollistettuna kuvassa 1 on valkoisten kivien vapaudet merkittynä rasteilla.



Kuva 1. Kivien vapaudet

ii) Ko-sääntö eli toistosääntö: yhdessä vuorossa kivellä ei saa koskaan toistaa edellisellä vuorolla pelattua koko laudan tilannetta, vaan kivi pitää pelata toiseen pisteeseen. Kuvassa 2 kohdassa (a) on alkutilanne, jossa mustan pelaajan kivi on valkoisen pelaajan kivien saartama. Kohdassa (b) valkoinen pelaaja vangitsee mustan kiven rastilla merkityssä kohdassa. Ko-sääntö estää mustaa pelaamasta kiveään rastilla merkittyyn kohtaan seuraavalla vuorollaan, jonka vuoksi musta pelaa toisaalle kohdassa (c).



Kuva 2. Ko-sääntö

Gossa käytetään japanilaisista budolajeista tunnettua luokitteluasteikkoa, jossa pelaajat luokitellaan taitojensa mukaan asteikolla 30 kyu - 1 kyu » 1 dan - 9 dan. Aloittelevan pelaajan taso on aluksi noin 30 kyu ja nousee tästä 1 kyun tasoon. Tästä pelaajan taso nousee arvolle 1 dan ja siitä on mahdollista nousta tasolle 9 dan [Paatero, 2008]. Ammatikseen pelaavia dan-arvolla olevia pelaajia kutsu-

taan myös pro-nimityksellä (esim. 5 pro) [Macfadyen, 1998]. Gossa annetaan tasoituskiviä luokitukseltaan heikommalle pelaajalle vahvemman ja heikomman pelaajan arvojen erotuksen verran, tai enemmänkin. Esim. tasoltaan 10 kyun pelaaja saa 5 kiveä laudalle 5 kyun pelaajaa vastaan.

2. Go ja tietokoneet

Suhteellisen yksinkertaisista säännöistä huolimatta gota pelaavien tietokoneohjelmien kehittäminen on ollut haasteellista. Tietokone-gon tutkimukseen on käytetty huomattavia määriä aikaa ja vaivaa. Gota pelaavat ohjelmat ovat siltikin kehityksessä jäljessä, muihin suosittuihin lautapeleihin verrattuna [Müller, 2001].

Tietokone-gohon liittyy monia varteenotettavia teknisiä, käsitteellisiä ja ohjelmistosuunnittelun ongelmia. Monet gota pelaavat ohjelmat ovat 5-15 henkilötyövuoden tulosta ja sisältävät 50-100 erilaista komponenttia, jotka käsittelevät pelin eri osa-alueita. Silti ohjelman suoriutumiskyky on yhtä vahva kuin sen heikoin komponentti [Müller, 2002].

2.1 Miksi go on vaikeaa tietokoneille?

Go on hyvin abstrakti peli. Kivien määrä ja kompleksisuus lisääntyvät, mitä pidemmälle peli etenee. On helppoa kehittää ohjelma, joka osaa pelata kokonaisen pelin gota, mutta on vaikeaa kehittää ohjelma joka osaa pelata gota hyvin [Müller, 2002]. Bagdisin [2007] mukaan gosta on havaittu neljä keskeistä seikkaa, jotka vaikeuttavat normaaleja pelitekoälyissä käytettyjä tekniikoita:

i) Pelin tila-avaruus on erittäin suuri.

ii) Pelipuun haarautuvuus on erittäin suuri.

iii) On vaikeaa luoda heuristinen funktio, joka päättelisi pelilaudan tilan pätevästi.

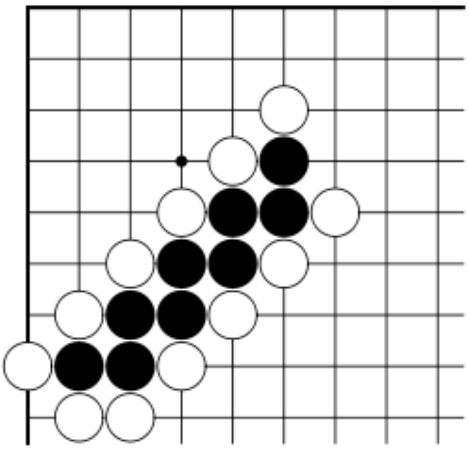
iv) Pätevä pelistrategia tarvitsee hyvin pitkälle ja syvälle suunniteltuja ja harkittuja siirtoja.

Normaalissa pelilaudassa on 361 pistettä. Jokainen näistä pisteistä voi itsenään olla kolmessa eri tilassa: pisteessä on valkoinen kivi, pisteessä on musta kivi tai piste on tyhjä. Tämä määrää 3^{361} erilaista tilaa pelilaudalle, joka on melkein 10^{172} [Bagdis, 2007]. Näistä eri tiloista n. 1,2% on pelille laillisia tiloja [Müller, 2002]. Melkein jokaisessa pelitilanteessa tyhjät pisteet ovat laillisia liikkeitä ja pelin alkutilanteissa laillisia siirtoja on yli 350. Laillisten siirtojen määrä vähenee pelin edetessä, mutta silti näiden siirtojen määrä on keskimäärin 200. Tästä johtuen eri pelitilanteiden määrä laudalla on huomattavan suuri, ja siksi on erittäin vaikeaa arvioida siirron tärkeyttä ja painoarvoa. Kun jokainen kivi

on keskenään saman arvoinen, niin esimerkiksi pelaajan kivien määrän laskeminen on hyödytön tapa arvioida siirtojen vahvuutta ja onnistumista eri tilanteissa [Bagdis, 2007]. Verrattuna samankaltaisiin lautapeleihin, jotka voivat käyttää keskenään samoja arviointimenetelmiä, gota varten on pitänyt kehittää uusia menetelmiä ja malleja, jotka vastaisivat gon huomattavaan kompleksisuuteen [Bouzy and Cazenave, 2001].

2.2 Ihmisen tietämys gossa

Ihmiset, jotka pelaavat ammattilaistasoilla gota, osaavat tunnistaa hienovaraisimmistakin siirroista, miten ne tulevat vaikuttamaan pelin myöhemmässä vaiheessa, ja osaavat pätevästi arvioida pienten ja isojen kiviryhmien vahvuudet, jopa pelin alkuvaiheilla. Tämä taito on olennainen kivien vahvuuden hahmottamiseksi. Samanlaisen tietämyksen hahmottaminen tietokonehaulla vaikuttaa miltei mahdottomalta. Ihmiset osaavat myös tilanteenhallinnan ja hahmottamisen paljon paremmin. Peliä paljon pelanneet voivat arvioida, kumpi puoli on vahvempi, ja osaavat arvioida lopullisen pistemäärän nopealla katsauksella pelilautaan. Koska kivet eivät liiku pelilaudalla siirron jälkeen, on ihmisten helpompi hahmottaa nykyisia ja tulevia siirtoja verrattuna muihin peleihin, esim. shakkiin. Gossa jopa muutaman pelin pelannut amatööripelaaja voi jo hahmottaa 15-20 siirtoa eteenpäin vaikeassa paikallistaistelussa, tai jopa 50 siirtoa tai enemmän, jos kyseessä on gossa tunnettu tikasmuodostelma (kuva 3) [Müller, 2002].



Kuva 3. Tikasmuodostelma

3. Tietokone-gon historiaa

Ensimmäisen gota pelaavan tietokoneohjelman kehitti D. Lefkovitz 1960-luvulla ja ensimmäinen tieteellinen julkaisu tietokone-gosta julkaistiin 1963. Siinä pohdittiin koneoppimisen hyödyntämistä gon peluussa. A. Zobrist kehitti ensimmäisen gota pelaavan ohjelman, joka voitti juuri pelin säännöt oppineen ihmisen. Tämä ohjelma hyödynsi funktiota, joka likimäärin arvioi kivien keskinäisen vaikutuspiirin pelilaudalla [Bouzy and Cazenave, 2001]. Ensimmäiset gota pelaavat ohjelmat pohjautuivat yksinomaan funktioihin, jotka arvioivat kivien vaikutuspiiriä; jokainen kivi hehkuu omaa vaikutuspiiriään läheisiin risteyksiin ja hehkun voimakkuus vähenee kivien välimatkojen kasvaessa. Vastapuolen kivien hehkun arvot ovat toistensa käänteisfunktioita. Näitä funktioita käytetään nykyisissäkin go-tekoälyissä [Bouzy and Cazenave, 2001]. Bruce Wilcox kirjoitti ensimmäisen gota pelaavan ohjelman, joka pelasi paremmalla tasolla kuin juuri säännöt oppinut. Wilcox kehitti peliä varten uuden teorian, jossa pelilauta jaettiin erillisiin alueisiin (sektoreihin). Tästä seuraava läpimurto tekoälyille oli hahmotuksen ja pelimallien lisääminen, joiden avulla tietokone tunnisti yleisempiä pelitilanteita. Kaikki nykyiset go-tekoälyt käyttävät näitäkin tekniikoita hahmontunnistuksesta abstrakteihin tietorakenteisiin. 1980-luvulla tietokone-go teki suuren harppauksen eteenpäin, ja tietokone-gosta tuli aivan oma tutkimuksen alansa ja mm. tekoälyjen välisiä kilpailuja ryhdyttiin järjestämään [Bouzy and Cazenave, 2001]. Tavalliset PC:t halpenivat, ja eri säätiöt ryhtyivät sponsoroimaan tekoälyjen välisiä kilpailuja (mm. Ing-säätiö) [Müller, 2002]. Myös ensimmäiset kaupalliset gota pelaavat ohjelmat julkaistiin 1980-luvulla. Seuraavalla vuosikymmenellä gota pelaavien tietokoneohjelmien määrä kasvoi huomattavasti, ja ohjelmien välisissä kisoissa nähtiin parhaimmillaan 40 osallistujaa [Bouzy and Cazenave, 2001].

Tietokoneiden ja laskinten yleistyessä, on go-kilpailuissa ryhdytty käyttämään myös perinteisen luokitteluasteikon sijaan Elo-arvostelulukuja [Wikipedia, 2012b]. Elo-luku on Arpad Elon kehittämä suhdeluku, jota käytetään kahden pelaajan peleissä mittamaan pelaajien välistä tasoeroa [Wikipedia, 2012c]. Elo-arvostelulukujen laskeminen vaihtelee eri go-kilpailuiden ja -yhteisöjen kesken, mutta yksi yleisimmistä on European Go Federationin määrittelemä laskutapa. Tässä määrittelyssä voidaan antaa perinteiselle luokittelutasolle vastaava Elo-luku. Esimerkiksi perinteistä 20 kyun tasoa vastaa Elo-luku 100 ja yhden danin tasoa vastaa Elo-luku 2100 [Wikipedia, 2012b].

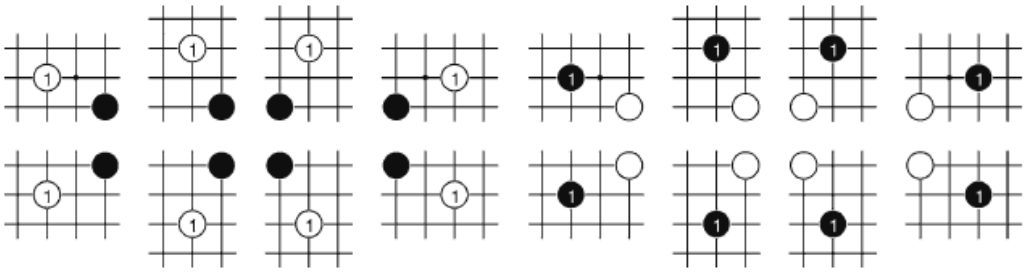
4. Go-tekoälyjen komponentteja

On olemassa monia eri tapoja mallintaa go-tietämystä. Ohjelmaan voidaan sisällyttää jo tiedettyjä pelikuvioita ja malleja, käsin tai ohjelmallisesti. Toinen yleinen menettely on luoda jokin rakenteellisempi esitys käyttäen hierarkisessa järjestyksessä olevia komponentteja ja näin luoda ohjelmalle korkeamman tason tietämystä pelin tilasta [Müller, 2002]. Tälläinen oppimisjärjestelmä vähentäisi käsinohjelmoidun tietämyksen määrää, joka muuten lisäisi ohjelman kompleksisuutta ja tekisi siitä vaikeasti parannettavan [van der Werf et al., 2011].

4.1 Pelikuviot ja kuvioiden hahmotus

Pelikuviot ovat tehokas keino lisätä tietämystä tekoälyyn. Monet gon tilanteet kuten *josekit* (pelilaudan kulmissa tapahtuvat tilanteet) ja *tesujit* (yksittäisiä taktisesti vahvoja siirtoja) perustuvat hyväksi havaittuihin kuvioihin. Kuvioita voi käyttää pelin jokaisessa vaiheessa alusta loppuun ja melkein jokaisella gota pelaavalla ohjelmalla on tietokanta kuvioista sekä jonkinlainen järjestelmä niiden hahmottamiseen [Müller, 2002]. Perinteiset gota pelaavat ohjelmat käyttävät näitä tietokantoja pätevien liikkeiden arvioinnissa. Monesti nämä kuviotietokannat ovat isoin ja pitkäkestoisin osa tekoälyn kehitystyötä [Gelly and Silver, 2011].

Gossa yhtä koko laudan pelitilannetta pitää verrata pelikuvioon kuudellatoista eri tavalla (havainnollistettu kuvassa 4), eli siis jokainen kuvio voi ilmentyä pelissä eri paikoissa kahdeksalla eri tavalla mustilla ja valkoisilla kivillä. Tämä vaatisi hyvin paljon laskemista tietokoneelta, jos jokaisella siirrolla verrataan jokaista laillista laudan pistettä jokaiseen pelikuvioon. Ongelmaan on kehitetty suodattavia hakumenetelmiä, jotka perustuvat hajautustauluihin ja trietietorakenteisiin. Nämä hakumenetelmät vähentävät huomattavasti siirtoon verrattavien pelikuvioiden määrää [Müller, 2002].



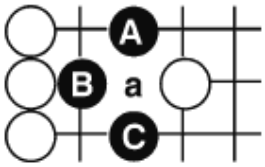
Kuva 4. Pelikuvion 16 erilaista instanssia. [Müller, 2002]

4.2 Tietämyksen luominen go-pelistä

Gota pelaavat ohjelmat sisältävät komponentteja, jotka mallintavat gon käsitteitä eri abstraktion tasoilla. Nämä abstraktiot eroavat hyvin paljon ihmispelaajille suunnatuista esityksistä, ja ne soveltuvat paljon paremmin tietokoneen prosessoinnille [Müller, 2002]. Müller [2002] on listannut muutamia näitä komponentteja:

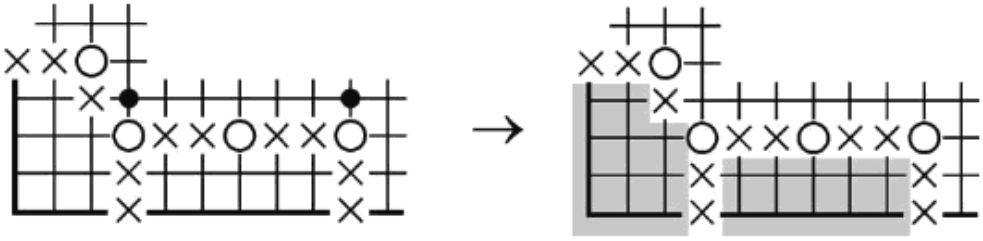
- *Kirjastot.* Tekoälyn pohjalla on hyvä olla jokin rajapinta tai kirjasto, joka tukee korkeamman tason abstraktioiden implementoimista. Esimerkkinä STL-kirjasto C++-kielelle.
- *Go-lauta.* Lauta yleensä abstrahoidaan kokonaislukutaulukkona tai bittikarttana. Laudan implementointi on monesti samanlainen kuin muiden lautapelien laudan implementointi.
- *Siirrot ja säännöt.* Kaikista pelatuista siirroista on hyvä olla tietämys, mm. ko-säännön havaitsemiseen ja siirron peruuttamiseen. Pelatut siirrot on yleensä implementoitu pino-tietorakenteella.
- *Lohkot.* Müller [2002] nimeää yhdessä olevia kiviä lohkoiksi (eng. block). Lohkot ovat primitiivisiä peruselementtejä pelilaudan esityksessä ja niiden ominaisuuksiin kuuluvat kivet, vapaudet, yhteydet muihin ryhmiin ja yhteydet muihin tietorakenteisiin, kuten alueisiin.
- *Liitännät, jakajat ja alueiden linjat.* Lohkot voivat liittää tai erottaa muita kiviryhmiä keskenään sekä toimia alueen jakajina. Nämä jakajat voivat mm. estää vastapuolen pelaajaa laajentamasta aluettaan.
- *Ketjut.* Ketjut ovat liitännällä yhdistetty lohkoja. Monissa gota pelaavissa ohjelmissa ei tehdä erottelua ketjujen ja ryhmien välillä. Ketjujen määrittelyssä esiintyy yleisesti se ongelma, että gon liitokset eivät ole transitiivisia. Eli, jos A on liitoksissa B:hen ja B on liitoksissa C:hen, niin tämä ei

tarkoita sitä, että A olisi liitoksissa C:hen. Tätä on havainnollistettu kuvassa 5, jossa ketjuina ovat kivet {A,B} ja {B,C}, mutta ei {A,B,C}, sillä valkoisen kivi uhkaa ryhmää kohdassa 'a' ja voi katkaista A:n tai C:n muista kivistä.



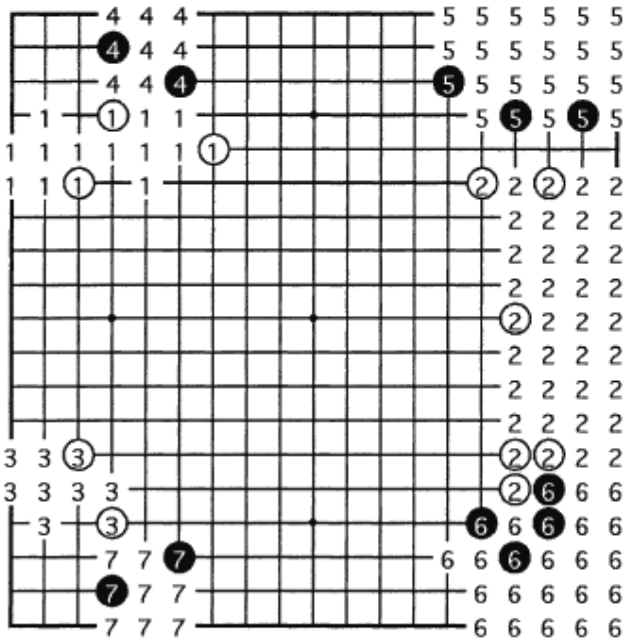
Kuva 5. Kiviuhka ja transitiivisuus [Müller, 2002]

- *Saarretut ja turvalliset alueet, kivien potentiaalit.* On erittäin tärkeätä havaita, mitkä alueet laudalla ovat pelaajan saartamia. Saarretuilla alueilla määritellään pelin voittaja. Alueet tarjoavat myös lisävapauksia ryhmille joita nimitetään 'silmiksi'. Alueet voidaan määritellä myös kivien potentiaalilla, eli niiden vaikutuspiirillä, tai löytämällä kivilohkoja ja jakajia, jotka erottavat alueita. Kuvassa 6 on kolme valkoisen pelaajan kiveä, joiden potentiaalinen alue on havainnollistettu tummalla värillä ja niiden jakajat rasteilla.



Kuva 6. Kivien potentiaalin havaitseminen jakajilla. [Müller, 2002]

- *Ryhmät.* Ryhmät sitovat edellämainitut käsitteet yhdeksi määritelmäksi. Gota pelaavat ohjelmat tunnistavat ryhmät yhtenäisiksi alueiksi, joilla on tietty vaikutuspiiri. Ryhmiä käytetään gossa puolustukseen ja hyökkäykseen. Ryhmien suhteellinen vahvuus ja vaikutusalue määrittelee, selviääkö ryhmä laudalla, tai voiko ryhmä esimerkiksi tukea muita ryhmiä hyökkäyksessä tai puolustuksessa. Tätä vahvuutta mitataan ryhmän sisäisillä ja ulkoisilla liitännöillä, ryhmän lohkojen vapauksilla ja saarretun alueen silmätilalla (vapauksien määrällä). Kuvassa 7 on Müllerin [2002] oman go-ohjelma Explorerin arviointi ryhmistä yhdestä pelitilanteesta. Kivien päällä ja tyhjissä pisteissä olevat numerot merkitsevät samaa ryhmää ja niiden potentiaalista vaikutusaluetta.



Kuva 7.
Explorer-ohjelman arvio ryhmistä. [Müller, 2002]

4.3 Muita ongelmia

On havaittu, että gon tiettyihin ongelmiin on ollut syytä kehittää omat erikoistuneet metodinsa ja alikomponenttinsa. Tunnetuin ongelma on ehkä kivien elämän ja kuoleman määritteleminen. On olemassa kaksi erilaista määritelmää kivien elämälle ja kuolemalle. Japanilaisissa säännöissä mainitaan: ”Kivet ovat elossa, jos vastustaja ei pysty vangitsemaan niitä, tai jos niiden vangitsemisen jälkeen voidaan pelata uusi kivi, jota vastustaja ei pysty vangitsemaan. Kivet, jotka eivät ole eläviä, ovat kuolleita.” Puolestaan kiinalaiset säännöt määrittele-

vät asian näin: ”Pelin lopussa ne kivet, jotka pelaajat määrittelevät vangituiksi, ovat kuolleita. Kivet, joita ei pysty vangitsemaan, ovat eläviä.” Nämä säännöt voivat aiheuttaa ristiriitoja pelitilanteen tulkinnassa, koska joissain tapauksissa kivet voivat olla elossa japanilaisten sääntöjen mukaan, ja kuolleita kiinalaisten sääntöjen mukaan [van der Werf et al., 2011].

4.4 Hakumenetelmät gossa

Gota pelaavissa ohjelmissa käytetään erilaisia hakumenetelmiä tiettyihin tehtäviin. Müller [2002] on erotellut kolme yleisintä hakupuuta: yhden tavoitteen haku, monen tavoitteen haku ja koko laudan haku. Vaikein näistä hakupuista on koko laudan haku, johtuen syistä joita on käyty läpi luvussa 2. Yleisesti tavoitepainotteinen haku on paljon nopeampi kuin koko laudan haku, koska ohjelma joutuu yleensä arvioimaan vain pienen osan tilanteesta. Tästä johtuen monia tavoitepohjaisia hakuja käytetään yhden globaalin siirron arviointiin.

Monte-Carlo -haku on uusi menetelmä pelihaulle, joka on mullistanut tietokone-gon. Tämä menetelmä on korvaamassa perinteisempiä hakualgoritmeja muillakin pelitutkimuksen aloilla [Gelly and Silver, 2011]. Monte-Carlo -menetelmän perusajatus on simuloida tuhansia satunnaisia pelituloksia sen hetkiselle pelitilanteelle. Uudet pelitilanteet lisätään hakupuuhun, ja hakupuun jokainen alkio sisältää arvon, josta voidaan ennustaa, kuka voittaa pelin kyseisestä pelitilasta. Eli siis jokaisen alkion arvo on keskimääräinen tulos kaikista simuloiduista pelitilanteista, jotka käyvät alkiossa pelin aikana. Monte-Carlo -simulaatio päivittää tätä hakupuuta pelin edetessä [Gelly and Silver, 2011]. Monte-Carlo -simulaatiota käyttävien ohjelmien Elo -luku nousee aina suorittimien määrän kasvaessa [Dailey, 2008]. Monte Carlo -simulaatiota käyttävillä ohjelmilla on heikkoutena ratkaista tilanteita, joissa liikkeiden järjestys on tärkeä [Wikipedia, 2012a].

5. Tietokone-gon nykytila

Tietokone-gon lyhyessä historiassa ihmispelaajat ovat olleet miltei voittamattomia. Kolmenkymmenen vuoden kehitystyön aikana go-tekoälyissä on saavutettu taso, joka vastaa ihmisten tasolla amatööripelaajaa [Gelly and Silver, 2011]. Vasta 2000-luvulla gota pelaavat ohjelmat ovat saavuttaneet ihmisiä pelitaidoissa. Uudet hakumenetelmät, kuten Monte-Carlo -menetelmä, ovat nostaneet ohjelmien tasoa huomattavasti. Esimerkiksi perinteisiin hakumenetelmiin perustuvat tekoälyt ovat Elo-arvostelutasolla noin 1800 ja Monte-Carlo -menetelmää käyttävät ovat tasolla 2500 (Isompi luku on parempi). Huomattavaa on, että yli 700 Elo-pisteen erotus tarkoittaa 99% voittomahdollisuutta [Gelly and

Silver, 2011]. Myös tietokoneiden halpeneminen ja niiden prosessointitehon nouseminen on edesauttanut gota pelaavien ohjelmien kehitystä. Tänä vuonna maaliskuussa japanilainen tekoäly nimeltään Zen voitti japanilaisen 9 pro tasolla olevan ammattilaisen kaksi kertaa peräkkäisissä otteluissa. Ensimmäisessä ottelussa Zen sai viisi kiveä tasoitusta ja voitti yhdellätoista pisteellä. Toisessa ottelussa Zen sai neljän kiven tasoituksen ja yllätti voittamalla kahdellakymmenellä pisteellä [Ormerod, 2012]. Gota pelaavat ohjelmat ovat saavuttamassa jo etulyöntiasemaa amatööripelaajia vastaan, mutta ammattilaispelaajat antavat vielä haastetta nykyisille tekoälyille.

5.1 Muutamia gota pelaavia tekoälyjä

GNU Go

GNU Go on ehkä tunnetuin avoimeen lähdekoodiin perustuva gota pelaava tekoäly. Ensimmäinen stabiili versio julkaistiin *comp.sources.games*-uutisryhmään 13. maaliskuuta 1989 ja viimeisin stabiili versio julkaistiin 19. helmikuuta 2009 [GNU Go, 2012a]. Ensimmäiset stabiilit versiot olivat luokitukseltaan 30-kyu, ja uusien versioiden kanssa luokitus on kasvanut jopa 5-kyuhun asti Kiseido Go -palvelimella [Sensei's Library, 2012]. GNU Gon Elo-luku on 1800 [Dailey, 2008]. GNU Go sisältää klassista go-tietämystä ja paljon käsinohjelmoituja kuvioita [Baudis, 2009].

GNU Gon dokumentaatiossa [2012b] on kerrottu yleisellä tasolla, miten tekoäly käsittelee yhden vuoron. Aluksi GNU Go yrittää ymmärtää laudan tilanteen niin hyvin kuin on mahdollista. Mainittakoon, että GNU Gossa ei ole koko laudan hakua implementoitu. Ensimmäisen vaiheen informaatiota hyväksi käyttäen luodaan lista mahdollisista kandidaattiliikkeistä. Lopuksi, jokainen näistä kandidaattiliikkeistä arvioidaan sen alueellisen ja strategisen arvon mukaan.

Zen

Zen on japanilaisen Yoji Ojiman ohjelmoima tekoäly [Ormerod, 2012]. Zen aloitti pelaamisen Kiseido Go-palvelimella, jossa se nousi luokitukseen 2-dan, tietokoneprosessorinaan neljäytiminen AMD Opteron [Wedd, 2012b]. Kuten aiemmin mainittu Zen voitti 2012 maaliskuussa japanilaisen 9 pro-ammattilaisen Takemiya Masakin ensin viiden ja sitten neljän kiven tasoituksella. Tässä ottelussa Zeniä suoritettiin neljän PC:n ryppäällä; ensimmäisessä tietokoneessa oli kaksi kuusiytimistä suoritinta ja toisessa oli yksi kuusiytiminen suoritin. Kolmannes sekä neljännes tietokone sisälsi kaksi neliytimistä suoritinta. Tietokoneet olivat yhteydessä keskenään yhden gigabitin lähiverkolla. Nykyään Zen pelaa Kiseido Go -palvelimella tällä samalla kokoonpanolla nimimerkein 'zen19s' ja 'zen19d' [Ormerod, 2012]. Zen käyttää Monte-Carlo -

menetelmiä, mutta sisältää myös paljon käsinohjelmoitua go-tietämystä [Wedd, 2012b]. Zen:in mullistavuus piilee sen vahvuudessa käyttää kuluttajasuuntaista tietokonelaitteistoa.

6. Yhteenveto

Tietokone-go on vieläkin haasteellista ohjelmoijille. Vaikka muutamat tekoälyt pärjäävät amatööritason ihmisille, niin ammattilaispelaajia vastaan ne tarvitsevat vieläkin tasoituskiviä. Voi mennä vielä pitkä aika ennen kuin tietokoneet osaavat pelata gota samalla tasolla kuin ne pelaavat shakkia. Feng-hsiung Hsu [2007] uskoo, että kymmenen vuoden sisään kehitetään gota pelaava ohjelma, joka on maailman parhaimpien go-mestarien tasolla puhtaasti raa'an laskentatehon vuoksi. Uudet menetelmät ovat nostaneet ohjelmien tasoa, mutta gota hyvin pelaava ohjelma tarvitsee mielestäni hyvän ja laajan liiketietokannan, kuin myös pätevän hakualgoritmin, muiden ominaisuuksien lisäksi. Gota pelaavien ohjelmien kehitystä on mielenkiintoista seurata, sillä ne ongelmat mihin nämä ohjelmat vastaavat voivat aukaista täysin uusia ratkaisuja muihinkin tietojenkäsittelyn ongelmiin. Ei ole siis ihmekään, että suomalaisissa go-kerhoissa on lentävänä lauseena: ”Kun tietokoneet oppivat pelaamaan gota, ne oppivat myös luomaan taidetta.”

Viiteluettelo

[Bagdis, 2007] Jeffrey Bagdis, A Machine-Learning Approach to Computer Go. Available as www.cs.princeton.edu/~jbagdis/jp.pdf. Checked 29.9.2012.

[Baudis, 2009] Petr Baudis, Current approaches in computer go. Available as http://pasky.or.cz/~pasky/go/compgo-r2.pdf. Checked 29.9.2012.

[Bouzy and Cazenave, 2001] Bruno Bouzy and Tristan Cazenave, Computer go: an AI oriented survey. *Artificial Intelligence* **132**, 1 (October 2001), 39-103.

[Dailey, 2008] Don Dailey, 9x9 scalability study. Available as http://cgos.boardspace.net/study/index.html. Checked 29.9.2012.

[Gelly and Silver, 2011] Sylvain Gelly and David Silver, Monte-Carlo tree search and rapid action value estimation in computer go. *Artificial Intelligence* **175**, 11 (July 2011), 1856-1875.

[GNU Go, 2012a] GNU Go, GNU Go Development Versions. Available as http://www.gnu.org/software/gnugo/devel.html. Checked 18.11.2012

[GNU Go, 2012b] GNU Go, GNU Go engine overview. Available as http://www.gnu.org/software/gnugo/gnugo_4.html. Checked 18.11.2012

[Hsu, 2007] Feng-hsiung Hsu, Cracking Go. *IEEE Spectrum* **44**, 50-55.

[Macfadyen, 1998] Matthew Macfadyen, *The Game of Go*. Carlton Books Ltd, 1998.

[Müller, 2001] Martin Müller, Review: computer go 1984-2000. *Lecture Notes in Comput. Sci.* **2063** (2001), 405-413.

[Müller, 2002] Martin Müller, Computer go. *Artificial Intelligence* **134**, 1-2 (January 2002), 145-179.

[Ormerod, 2012] David Ormerod, Zen computer go program beats Takemiya Masaki with just 4 stones! Available as http://gogameguru.com/zen-computer-go-program-beats-takemiya-masaki-4-stones/. Checked 1.10.2012.

[Paatero, 2008] Lauri Paatero, *GO – Mikä se on?*. Books on Demand GmbH, 2008.

[Sensei's Library, 2012] Sensei's Library, GNU Go. Available as http://senseis.xmp.net/?GnuGo. Checked 18.11.2012

[van der Werf et al., 2011] Erik C.D van der Werf, Mark H.M. Winands, H. Jaap van den Herik and Jos W.H.M. Uiterwijk, Learning to predict life and death from go game records. *Information Sciences* **175**, 4 (15 November 2005), 258-272.

[Wedd, 2012a] Nick Wedd, Human-Computer Go challenges. Available as http://www.computer-go.info/h-c/index.html. Checked 1.10.2012

[Wedd, 2012b] Nick Wedd, Human-Computer Go challenges. Available as http://www.computer-go.info/db/oprog.php?a=Zen. Checked 18.11.2012

[Wikipedia, 2012a] Wikipedia, Computer Go. Available as http://en.wikipedia.org/wiki/Computer_Go. Checked 1.10.2012

[Wikipedia, 2012b] Wikipedia, Go ranks and ratings. Available as http://en.wikipedia.org/wiki/Go_ranks_and_ratings. Checked 27.12.2012

[Wikipedia, 2012c] Wikipedia, Elo rating system. Available as http://en.wikipedia.org/wiki/Elo_rating. Checked 27.12.2012

Perinnejärjestelmän integrointi – Case Lapin keskussairaala

Markus Salomaa

Tiivistelmä.
Tietojärjestelmät ovat osa nykyaikaista liiketoimintaa ja erilaisten järjestelmien integraatiolla yritetään helpottaa tiedon saamista ja parantaa prosessien tehokkuutta. Kriittisissä käyttökohteissa tietojärjestelmien käyttöikä kuitenkin pitenee, jolloin integraatio uusien järjestelmien kanssa muodostuu hankalaksi. Tässä tutkielmassa käsitellään tietojärjestelmien integraatiota alan kirjallisuuden ja case-esimerkin kautta. Tutkielman painotus on terveydenhuollon tietojärjestelmissä, erityisesti terveydenhuollossa esiintyvissä perinnejärjestelmissä.

Avainsanat ja -sanonnat: tietojärjestelmäintegraatio, perinnejärjestelmät, terveydenhuollon tietojärjestelmät
CR-luokat: J.3

1. Johdanto

Tietojärjestelmät ovat osa meidän kaikkien jokapäiväistä elämää, sillä erilaista tietoa tuotetaan lukemattomiin julkisen ja yksityisen sektorin järjestelmiin. Kirjastot ylläpitävät lainaustietojärjestelmiä, jotka koostuvat muun muassa asiakkaista, lainaustietokannasta ja kirjaston henkilökunnasta. Lähikaupassa on käytössä tietojärjestelmä, joka koostuu yksittäisistä kassakoneista, maksupäätteistä ja tukkukauppiaaan verkkoon yhdistetyistä tietokannoista.

Tietojärjestelmän tarkoituksena on tietojen käsittelyn avulla tehostaa tai mahdollistaa jokin haluttu toiminta. Tietojärjestelmistä on tullut yksi keskeisimmistä tukipalveluista nykyaikaisessa kilpailutilanteessa olevalle yritykselle ja mahdollisimman hyvin saatavilla olevan tiedon arvo korostuu jatkuvasti. Terveydenhuollon sektorilla välittömästi saatavilla olevan tiedon tarve voi olla joskus hyvin kriittistä ja välttämätöntä, koska sillä voidaan turvata ihmishenkiä. Näin ollen tietojärjestelmien avulla voidaan tehostaa ja kehittää varsinaista ydinprosessia, potilaan hoitoa, jotta lopputuoteesta, terve potilas, saadaan kustannuksiin nähden mahdollisimman hyvä. Tiedon uudelleenkäytettävyydestä on tullut myös ydinkysymys, kun halutaan kehittää jatkuvasti parempia järjestelmiä.

Valitettavasti tämä kaikki voi olla vaikeasti saavutettavissa, sillä tieto voi olla liian hajallaan tai sitä ei ole kirjattu mihinkään järjestelmään. Tähän ongelmaan on haettu ratkaisua tietojärjestelmien integroinnin kautta, jonka avulla pirstoutunut tietoa voitaisiin kerätä kasaan siten, että se on ihmisten ja tietokoiden käytettävissä. Joskus voi syntyä tilanteita, jolloin integroitavan tietojärjestelmän ikä ja rakenne voi olla hyvin erilaisia verrat-

tuna muihin ympäristössä oleviin järjestelmiin. Tällöin yleensä puhutaan perinnejärjestelmän integroinnista.

Tässä tutkielmassa perehdytään perinnejärjestelmän integrointiin, kohdealuena terveydenhuollon tietojärjestelmät. Aluksi luvussa 2 esitellään tietojärjestelmiin kirjallisuudessa liitetyt keskeiset käsitteet ja niiden määritelmät, sillä arkikielessä tietotekniikan termit esiintyvät hyvin moninaisissa merkityksissä. Luvussa 3 käsitellään tietojärjestelmien integraation hyötyjä ja mahdollisia vaatimuksia. Näissä luvuissa käsitellään myös erilasia integraatiotapoja ja integraation mahdollisia sosiaalisia vaikutuksia. Lopuksi luvussa 4 perehdytään suomalaisen sairaanhoitopiirin todelliseen esimerkkiin perinnejärjestelmän integraatiosta nykyaikaisiin järjestelmiin. Luvussa tuodaan myös tarkemmin esille tarkastelun kohteena oleva vanhentunut tietojärjestelmä ja sen mahdollinen kohtalo seuraavan kolmen vuoden aikajaksolla. Luvussa käytettävät tiedot on hankittu työskentelemällä kyseissä sairaalassa ja haastattelemalla muita sairaalan työntekijöitä.

2. Tietojärjestelmä

Käsitteellä ”tietojärjestelmä” voidaan arkikielessä tarkoittaa monia asioita, jotkut ymmärtävät sen tietokoneohjelmaksi, toiset taas tiedonsiirtolaitteiden ja ohjelmistojen kokoelmaksi. Todellisuudessa rajaus ei kuitenkaan ole näin suppea, vaan tietojärjestelmällä on jo pitkään tarkoitettu kokonaisuutta, joka koostuu ihmisistä, tietojenkäsittelylaitteista, tiedonsiirtolaitteista ja ohjelmista. Tietojärjestelmien avulla on mahdollista tehostaa ja helpottaa toimintoja tai tehdä jotkut toiminnot ylipäätään mahdolliseksi. [Nunamaker and Briggs, 2011]

Tekniikan kehittyessä on tietojärjestelmistä tullut yhä olennaisempi osa organisaatioiden ja ihmisten elämää kaikilla sektoreilla. Tämän seurauksena myös käsitteet informaatioteknologia ja tietojärjestelmä sotketaan useasti keskenään. Informaatioteknologian perimmäisenä tarkoituksena käsitellä digitaalisia objekteja, kun taas tietojärjestelmien ideana on tuottaa käyttäjälleen arvoa. Näin ollen onkin esitetty, että tietojärjestelmien keskeisin tehtävänä on ymmärtää ja kehittää tapoja siitä, miten ihmiset luovat uutta arvoa informaation kautta. [Nunamaker and Briggs, 2011]

2.1. Perinnejärjestelmä

Perinnejärjestelmälle on olemassa lukuisia määritelmiä, mutta pääsääntöisesti sillä tarkoitetaan vanhentunutta tietojärjestelmää, joka on vielä käytössä, koska se on käyttöympäristössään elintärkeä [Bennett, 1995]. Perinnejärjestelmille on tunnusomaista korkea käyttöikä, sillä niitä käytetään sellaisissa käyttökohteissa, joissa järjestelmän uusiminen voi olla liian riskialtista tai kallista, kuten lentoliikenteessä. Koska järjestelmien käyttöikä on pitkä, niiden kehitys jatkuu aina siihen asti, kunnes vanha järjestelmä korvataan uudella. Tämän seurauksena perinnejärjestelmistä on voinut vuosien saatossa tulla hyvin monimutkaisia. Tällaiseen järjestelmän tilan kehittymiseen on sovellettu Lehmanin toista lakia, jonka mukaan ohjelmistoa kehittäessä kompleksisuus kasvaa, ellei sitä erikseen pyritä vähentämään

[Lehman, 1980]. Perinnejärjestelmät on yleensä kirjoitettu jo vanhentuneella ohjelmistokielellä ja ne ovat kalliita ylläpitää. Ylläpitokuluja nostavat järjestelmän eri osien mahdollinen eroavaisuus, kompleksisuus ja vanhentuneet ohjelmointikäytännöt. Toisaalta perinnejärjestelmät voidaan nähdä hyötynä ja tärkeänä resurssina, jota työntekijät ovat oppineet vuosien aikana käyttämään. [Sommerville, 2010]

Perinnejärjestelmät voidaan nähdä sosioteknisinä järjestelminä ja ne koostuvat yleisesti kuudesta komponentista. Järjestelmä toimii laitteistolla, joita ei mahdollisesti valmisteta tai se on kallista ylläpitää. Laitteistotaso määrää myös käytettävän käyttöjärjestelmän ja siinä olevat tukiohjelmistot, jotka voivat olla olennaisia itse pääjärjestelmän kannalta. Varsinainen ohjelmisto koostuu yleensä useasta pienemmästä ohjelmasta, jotka toteutettu eri aikoina ja sitten liitetty toimimaan yhdessä. Ohjelmistolle on olemassa myös jonkinlainen tietokanta, jossa voi olla suuri määrä kerääntynyttä tietoa vuosien varrelta. Tämän lisäksi perinnejärjestelmään kuuluu toimintaympäristön prosesseja, jotka on mahdollisesti aikoinaan suunniteltu perinnejärjestelmää silmälläpitäen. Yrityksen iskostuneet käytännöt ja säännöt voivat myös sisältää viittauksia tiettyyn perinnejärjestelmään. [Sommerville, 2010]

Perinnejärjestelmille on tyypillistä, että niistä eroon pääseminen on hyvin riskialtista ja hankalaa. Kuten edellä mainittua, järjestelmät toimivat yleensä sellaisissa käyttökohteissa, joissa riskien mahdollisuus pitää minimoida. Vanhoja tietojärjestelmiä ei välttämättä ole määritelty, vanhat dokumentaatiot ovat kadonneet tai niitä ei ole muistettu päivittää järjestelmäpäivitysten myötä. Näin ollen uutta järjestelmää on hankala määritellä vastaamaan vanhan järjestelmän toiminnallisuuksia. Onkin esitetty pääsääntöisesti neljää strategista vaihtoehtoa, hylkäys, ylläpito, uudelleen ohjelmointi tai järjestelmän korvaus, joita voidaan soveltaa perinnejärjestelmien kohdalla. Hylkäystä suositellaan silloin, kun järjestelmä ei tuota arvoa yrityksen prosesseihin, ylläpitoa taas silloin kun järjestelmä on vielä tarpeellinen ja toimiva. Uudelleen ohjelmointi voi tulla kyseeseen silloin, kun järjestelmän laatu on laskenut, mutta kehittäminen on ylipäätään mahdollista. Täysin uudella tietojärjestelmällä korvaaminen on suositeltavaa silloin, kun uusia toiminnallisuuksia ei voida toteuttaa järkevillä kustannuksilla. [Sommerville, 2010]

2.2. Terveydenhuollon tietojärjestelmät

Sairaalatietojärjestelmiä on ollut käytössä sairaaloissa noin 30 vuoden ajan, mutta viime vuosina nämä järjestelmät ovat kehittyneet osaksi terveydenhuollon tietojärjestelmiä. Käsitteellä terveydenhuollon tietojärjestelmä tarkoitetaan laajempaa kokonaisuutta, jonka tehtävänä on hallita ja tuottaa tietoa terveydenhuollon ympäristössä. Tämän kokonaisuuden avulla pystytään tarjoamaan potilaille laadukasta ja tehokasta hoitoa. Näin ollen sairaalatietojärjestelmät ovat vain osasia terveydenhuollon tietojärjestelmistä, siinä missä lääkärit ja sairaanhoitajatkin. [Haux, 2006]

Tietojärjestelmät ovat merkittävä osa nykyaikaista terveydenhoitoa, sillä valtaosa tiedosta on pystytty siirtämään paperilta sähköiseen muotoon. Saatavilla oleva tiedon määrä on myös kasvanut räjähdysmäisesti samalla, kun uusia hoitomuotoja on kehitetty. Tämän

varjopuolena on syntynyt tilanne, jossa laadukkaan hoidon antaminen on muuttunut lähes mahdottomaksi, jos hoitohenkilökunnalla ei ole käytettävissä toimivia tietojärjestelmiä. [Haux, 2006]

Terveydenhuollon tietojärjestelmille on tyypillistä, että ne jakautuvat moniin osa-alueisiin, kuten esimerkiksi potilastieto-, laboratorio- ja kuvantamisjärjestelmiin [Kuhn and Giusse, 2001]. Näiltä kaikilta vaaditaan korkeaa laatua, kustannustehokkuutta ja mahdollisuutta käsitellä suuria määriä dataa potilaiden turvallisuutta vaarantamatta. Vuosikymmenten aikana järjestelmien ja erilaisten tekniikoiden laaja kirjo on aiheuttanut sen, että tietojärjestelmien integroiminen on vaikeutunut, sillä yhtenäisiä standardeja ei ole kehitetty vaadittavalla tavalla. Integroitumisen puuttuminen on johtanut kustannustehokkuuden laskuun ja ylläpitokulujen kasvuun. Tätä tilannetta on kuitenkin yritetty parantaa ottamalla käyttöön standardeja, jonka avulla nykyaikaisten tietojärjestelmien on pystyttävä jakamaan tietoa toisten järjestelmien kanssa. Terveydenhuollon tietojärjestelmiin kohdistuu myös enemmän rajoituksia voimassa olevista lakipykälistä, kuin vastaaviin järjestelmiin muissa sektoreissa. Näiden säännösten avulla pyritään ylläpitämän potilaiden turvallisuutta ja rajoittamaan ulkopuolisten pääsyä kriittisiin potilastietoihin [Saboniiha *et al*., 2012].

3. Tietojärjestelmäintegraatio

Tietojärjestelmäintegraatiolla tarkoitetaan yleensä toimintaa, jossa organisaation sisälle hajautunut tieto yhdistetään ja näin aikaisemmin keskenään yhtyeensopimattomat sovellukset saadaan kommunikoimaan toistensa kanssa. [Tähtinen, 2005]. Sairaalaympäristössä tämä voi tarkoittaa esimerkiksi potilastietojärjestelmän ja röntgenkuvajärjestelmän yhdistämistä siten, että tieto liikkuu järjestelmien välillä. Tämän ansiosta järjestelmistä syntyy hallittava kokonaisuus. Näin ollen järjestelmäintegraatio ei ole yksittäinen tuote, vaan ajattelutapa, jonka avulla pystytään luomaan organisaatiolle mahdollisimman tehokas toimintaympäristö.

Järjestelmäintegraation historia juurtaa juurensa 1950-luvulle, jolloin käynnistettiin projektit SAGE (Semi-Automatic Ground Environment) ja SABRE (Semi-Automatic Business Research Environment). SAGE oli Yhdysvaltojen ilmavoimien projekti, jonka tarkoituksena oli kehittää automatisoitu ilmapuolustusjärjestelmä kylmän sodan tarpeisiin. SABRE taas oli American Airlines -lentoyhtiön kehittämä järjestelmä, jonka avulla hoidettiin lentojen paikanvaraus ja lipunmyynti. Kumpaakin projektia jatkokehitettiin, ja ne synnyttivät lukuisia vastaavia projekteja eri yrityssektoreille. SABRE-järjestelmä on yhä lentoyhtiöiden käytössä ja se on yksi suurimmista tietojärjestelmistä ilmailualalla [Tähtinen, 2005].

Tietojärjestelmien integrointi on järjestelmästä riippumatta kohdealueelleen tärkeä projekti ja isojen terveydenhoitojärjestelmien kohdalla se voi viedä useita vuosia. Tämän vuoksi integraation suunnittelu pitää tehdä huolella ja kauaskantoisesti. Lopputuloksen

kannalta on tärkeää tehdä heti alusta selväksi, miltä kannalta integraatiota lähestytään, sillä se voi tapahtua esimerkiksi organisatorisesta tai teknillisestä näkökulmasta. Integraatio voi tapahtua myös joko horisontaalisesti tietyn hierarkiatason sisällä tai vertikaalisesti eri hierarkiatasojen välillä. Perinteisesti integraatio on jaettu vielä useampiin kerroksiin, kuten data-, viesti- ja prosessitasoon, jotta kokonaisuudesta tulisi helpommin hahmotettava [Mykkänen *et al*., 2003].

3.1. Integraation hyödyt

Tietojärjestelmän ja sen integroinnin hyödyllisyyttä voidaan kuvata ns. Metcalfen lailla, jonka mukaan verkon arvo on verrannollinen verkon käyttäjien lukumäärän neliöön [Tähtinen, 2005]. Lain perusteella voidaankin päätellä, että tietojärjestelmäverkko, jossa on kuusi erillistä järjestelmää, on yhdeksän kertaa arvokkaampi kuin verkko, jossa on vain kaksi järjestelmää.

Terveydenhuollon sektorilla potilaiden tietoja siirretään ja tallennetaan useaan järjestelmään ja tietokantaan. Yhdelle potilaalle voidaan tehdä kymmeniä erilaisia tutkimuksia potilaan elinaikana, ja näiden hoitojen tulokset vaikuttavat jatkossa tapahtuvaan hoitoon. Jos keskeiset järjestelmät on integroitu toimivaksi kokonaisuudeksi, ei hoitohenkilökunnan tarvitse manuaalisesti syöttää tietoa kuin yhteen järjestelmään, jonka jälkeen tieto on myös muiden järjestelmien saatavilla. Jatkossa syötetty tieto on nähtävissä myös muista järjestelmistä, joten hoitohenkilökunnan aikaa ei kulu ylimääräisen tiedon etsintään, vaan työpanos voidaan keskittää varsinaiseen ydinprosessiin, eli potilaiden hoitoon. Potilaalle tämä näkyy parempana palveluna, ja jopa potilasturvallisuutta parantavana tekijänä, sillä potilaan ajan tasalla olevat tiedot ovat hoitohenkilökunnan saatavilla paikasta ja ajasta riippumatta. Potilaiden hoidot voidaan aloittaa nopeammin, kun tiedot on välittömästi saatavilla muista järjestelmistä. Järjestelmiin syötetty data on myös sisällöltään yhtenäistä, eikä näin ollen tulkintaeroja pääse syntymään niin helposti kuin erillisiä järjestelmiä käytettäessä. Esimiehet hyötyvät integraatiosta siten, että tieto on helpommin saatavilla ja näin ollen organisaatio on helpommin johdettavissa.

Kahden järjestelmän välille tehty erillisratkaisu on useasti vain väliaikaisratkaisu ja aiheuttaa turhaan eri liittymien määrän kasvua. Tämän vuoksi pitkäjänteinen integraatiosuunnittelu on toivottavaa heti alusta lähtien, sillä kasvava verkko luo myös kiinnostusta verkon ulkopuolella oleville palveluilla, ja verkko voi kasvaa suunnittelua nopeammin. Joustava tietojärjestelmä tarjoaa mahdollisuuden toimintaympäristön prosessien tai organisaation muutoksiin, sekä vähentää riippuvuutta ohjelmistotoimittajista [Tähtinen, 2005]. Kuten edellä mainittiin, järjestelmien integraatiolla pystytään jalostamaan kerran syötettyä tietoa, jonka avulla prosessia voidaan paremmin hallita. Tämä tukee päätöksentekoa ja antaa johdolle mahdollisuuden saada kokonaisnäkemys organisaation tilasta, muun muassa raportointityökalujen kautta. Hyvin suunniteltu tietojärjestelmäintegraatio selkeyttää myös tietojärjestelmäarkkitehtuuria ja näin muun muassa vähentää ylläpitokuluja.

Onkin käynyt ilmi, että ilman järjestelmäintegraatiota ei ole ollut mahdollista muodostaa laajempia tietojärjestelmäkokonaisuuksia. Tähän ei ole tullut muutosta, vaikka www-tekniikat ovat kehittyneet viime vuosina isoin harppauksin. Tiedon jakamisen tarve on yhä kasvanut, ja organisaatioiden tietojärjestelmien välisestä tiedonsiirrosta on tullut entistä merkittävämpää eri toimintaympäristöjen prosesseille. Esimerkkinä voidaan käyttää tulevaa terveydenhuollon eArkistoa, jonka tarkoituksena on luoda koko maan kattava sähköinen potilasarkisto. Sen avulla terveyskeskukset, sairaalat ja yksityiset terveysasemat voivat saada ulkopaikkakunnalta saapuneen potilaan hoitotiedot reaaliaikaisesti käyttöönsä, jotta asiakasta voidaan palvella mahdollisimman nopeasti ja hyvin. Vastakohtana voimme pitää vielä voimassa olevaa järjestelmää, jossa ei ole käytössä koko maassa toimivaa keskustietokantaa. Tämän seurauksena työntekijöiden työtehokkuus laskee eikä potilas saa parasta mahdollista hoitoa, koska kaikkea mahdollista tietoa ei välttämättä ole saatavilla tai ne joudutaan erikseen tilaamaan potilaan kotipaikkakunnalta.

3.2. Integraation vaatimukset

Kuten aikaisemmin on todettu, tietojärjestelmien integrointi ei tapahdu sormia napsauttamalla, vaan se voi olla hyvin pitkä ja vaikea prosessi erityisesti, jos integraation kohteena on esimerkiksi vaikeaselkoinen perinnejärjestelmä. Integraation perusedellytys kuitenkin on, että kahden järjestelmän välillä on edes jonkinlaiset rajapinnat, joita voidaan hyödyntää. Tämän lisäksi järjestelmien välillä tarvitaan jonkin fyysinen tiedonsiirtokanava, kuten tietoverkko tai internet [Tähtinen, 2005]. Harvasta järjestelmä on kuitenkaan niin suljettu, että integraatio olisi teknillisesti täysin mahdotonta.

Näin ollen voidaan sanoa, että integraation päävaatimus on järjestelmien yhtyeentoimivuus. Käsitteenä yhtyeentoimivuus voidaan jakaa vielä pienempiin osakokonaisuuksiin, jotka yhdessä muodostavat kokonaissuunnitelman eli kokonaisarkkitehtuurin. Saboniiha ja muut [2012] jakavat sairaalatietojärjestelmien yhtyeentoimivuuden seitsemään osatekijään, jotka muodostavat organisaatiorakenteen kanssa erikokoisia tasoja. Teknologisella näkökulmalla tarkoitetaan teknillisiä integraation lähtökohtia, joka voivat olla tiedon välitystä, tietoturvallisuutta ja tiedon tallennusta. Syntaktisella yhtyeentoimivuudella tarkoitetaan tiedon siirtoa järjestelmien välillä, kun taas rakenteellinen yhtyeentoimivuus koskee yleisesti hyväksyttyjä malleja ja käsitteitä, jotka mahdollistavat tiedon siirron. Semanttisen yhtyeentoimivuuden avulla välitettävästä tiedosta tulee integraation osapuolille ymmärrettävää ja mielekästä, kun taas operatiivisella yhtyeentoimivuudella tarkoitetaan sitä, miten hallinollista ja tilastollista tietoa pitäisi tulkita. Organisaation yhtyeentoimivuudella tarkoitetaan suurempaa kokonaiskuvaa erilaisista rooleista ja käytännöistä, siitä miten integroitavaa tietoa käytetään organisaation sisällä.

Näiden asioiden yhdistämistä varten on kehitetty lukuisia erilaisia lähestymistapoja ja integraatiomalleja. Näiden ideana on kerätä, dokumentoida ja hyväksikäyttää yleiskäyttöisiä malleja, joita voitaisiin soveltaa käyttökontekstissa. Terveydenhuollon järjestelmiin tällaiset mallit eivät kuitenkaan ole soveltuneet kovin hyvin, sillä sairaalajärjestelmät ovat

hiljalleen kehittyneet usean käyttäjäryhmän tarpeisiin. Tämän vuoksi järjestelmien kirjosta on tullut laaja, ja jäljelle jääneet perinnejärjestelmät ovat vielä elintärkeitä käyttökohteessaan. Joitain standardeja on kuitenkin ruvettu kehittämään juuri terveydenhuoltoa silmällä pitäen, näistä tunnetuimpana HL7 (Health Level 7), DICOM (Digital Imaging and Communication in Mediacal) ja EDIFACT (Electornic Data Interchange For Administration Commerce and Transport) [Beyer *et al*., 2004].

Tämän lisäksi myös yleiset ISO-standardit ovat edesauttaneet tietojärjestelmien kehitystä. ISO (International Organization for Standardization) on kansainvälinen standardisoimisjärjestö, joka tekee tietotekniikan osa-alueella yhteistyötä IEC:n (International Electrotechnical Commission) kanssa. Näiden kahden järjestön yhteistyönä on syntynyt muun muassa ISO/IEC 27000 -standardiperhe, jonka tarkoituksena on standardisoida tietoturvallisuutta ja sen hallintaa. Standardiperheestä löytyy myös ISO 27799 -standardi, joka on erityisesti suunnattu terveydenhuollon tietojärjestelmiä varten. Näillä määrityksillä on keskeinen rooli terveydenhuollon tietojärjestelmien suunnittelussa ja kehityksessä, sillä nykyaikana tietojärjestelmiltä useasti vaaditaan ISO-standardit täyttävät vaatimukset.

3.3. Erilaisia integraatiotapoja

Erilaisia integrointitapoja on lähes yhtä paljon kuin on erilaisia arkkitehtejäkin. Toteutukset vaihtelevat myös sen mukaan, millä integraatiotasolla aiotaan toimia. Helpoiten hahmotettava integraatiotapa on viestinvälitys, jonka perusideana on jakaa tietoa integroitavien järjestelmien välillä. Viestinvälitys on tehokas tapa, jolla voidaan suhteellisen helposti hoitaa kahden järjestelmän ”perinteinen” integraatio. Viestinvälitykseen on kehitetty lukuisia erilasisia standardeja, kuten HL7, DICOM ja XML. Viestinvälitys voidaan hoitaa usealla erilaisella käytännöllä, esimerkiksi järjestelmillä voi olla yhteinen jaettu tietokanta. Tietokannan lisäksi integraatio voidaan toteuttaa sillan, koordinaattorin tai väyläyhteistoiminnan avulla. [Saboniiha *et al*., 2012]

Siltayhteistoiminnan ideana on, että tieto siirretään ”sillan” kautta suoraan järjestelmästä toiseen [Mykkänen *et al*., 2003]. Tämän mallin huonona puolena on, että jos järjestelmät eroavat toisistaan, niin joudutaan käyttämään erityisistä ohjelmistoskriptejä ja koodinmuuntimia, jolla viestit pystytään kääntämään kohdejärjestelmän ymmärtämään muotoon. Tämän seurauksena järjestelmästä tulee vaikeasti ylläpidettävä ja tietoverkon rakenteesta hyvin monimutkainen. Virheiden löytämisestä tulee haastavaa, koska järjestelmiin saapuvat viestit käännetään aina erillisen ohjelmistokoodin avulla. Terveydenhuollon perinnejärjestelmissä on käytetty kyseistä toimintatapaa, sillä menetelmä voi olla lyhyellä aikavälillä kustannustehokas, mutta varsinaiset ongelmat esiintyvät ylläpito- ja laajennusvaiheessa.

Koordinaattori taas perustuu siihen ajatukseen, että kummankin järjestelmän yläpuolella on erillinen sovellus, jonka avulla järjestelmät voivat kommunikoida keskenään [Mykkänen *et al*., 2003]. Tämän hyvänä puolena on se, että järjestelmien ei tarvitse tietää tai osata keskenään toistensa protokollia, jotta integraatio olisi mahdollinen. Integraatiosta

tulee myös paremmin hallittu, sillä tiedetään tarkasti, missä osassa sovellusta toiminta tapahtuu.

Väyläyhteistoiminnan vaatii toimiakseen, että integroitavilla järjestelmillä on keskenään samanlainen infrastruktuuri, jonka tietyn tason sisällä tieto liikkuu [Mykkänen *et al*., 2003]. Tällaista on hyvin vaikea toteuttaa perinnejärjestelmiä integroitaessa, mutta jos integraatio on mahdollinen, voidaan saman väyläyhteistoiminnan avulla integroida useampia sovelluksia yhtä aikaa

Tietojärjestelmiä voidaan integroida myös sovelluskeskeisesti [Saboniiha *et al*., 2012]. Tällöin sovellusten yläpuolelle luodaan taso, jonka tehtävänä on välittää viestejä ja hallita integroitavia sovelluksia ja prosesseja. Perinteisestä viestinvälityksestä tämä eroaa siten, että tason kautta välittyy myös ohjelmistojen toimintalogiikka. Sovelluskeskeinen integraatio on vaikeampi toteuttaa esimerkiksi perinnejärjestelmien kanssa, koska vanhojen sovellusten toimintalogiikka on tunnettava hyvin. Terveydenhuollossa tämän integrointitavan päätarkoituksena on varmistaa, että viimeisin tieto on jatkuvasti saatavilla ja se on oikeanlaista.

Kolmas mahdollinen integrointitapa on koordinoitu integrointi, joka on suunniteltu enemmän loppukäyttäjän näkökulmasta [Saboniiha *et al*., 2012]. Integraatio voidaan toteuttaa esimerkiksi käyttäjän tiekoneelle työpöytäintegraation muodossa, jolloin käyttäjän tilaa synkronoidaan jatkuvasti ja tietoa lähetetään integroitavien järjestelmien ja ohjelmien välillä. Koordinoidun integraation avulla voidaan luoda myös niin sanottu kertakirjautuminen, jonka ideana on, että käyttäjälle riittää yksi käyttäjätunnus jokaiseen integraation alaiseen järjestelmään. Kertakirjautuminen on ollut terveydenhuollossa keskeinen puheenaihe, sillä tällä hetkellä lääkärit saattavat joutua kirjautumaan kymmeniin erilaisiin tietojärjestelmiin eri tunnuksilla.

Aikaisemmin mainittujen lisäksi on myös väliohjelmistoihin perustuva integrointi [Saboniiha *et al*., 2012]. Väliohjelmisto (middleware) on palvelinkoneella toimiva sovelluspalvelin, jonka tarkoituksena on hoitaa käyttöliittymän ja tietokannan yhdistäminen. Väliohjelmiston avulla käyttäjä voi lukea ja tallentaa useaan tietokantaan samanaikaisesti, jolloin käyttäjälle saadaan hallittu kokonaisuus, samalla tapaa kuin koordinoidussa integroinnissa. Väliohjelmiston etuna on myös, että eri järjestelmien väliohjelmistot voivat keskustella keskenään, ilman tietokantojen välistä suoraa linkkiä. Palveluita pystytään liittämään toisiinsa useiden tekniikoiden avulla kuten SOAP, Web Service ja CORBA. Perinnejärjestelmien kannalta väliohjelmistointegraatio on vaikeampi toteuttaa, sillä tarvitaan muutoksia jotta palvelut voivat liittyä yhteiseen infrastruktuuriin.

Teknisten ratkaisujen lisäksi voidaan käyttää myös semanttiseen yhteistoiminnallisuuteen perustuvaa integraatiota. Semanttisen yhteentoimivuuden ideana on, että tietojärjestelmä pystyy yhdistelemään ja käsittelemään eri tietolähteistä vastaanottamaansa tietoa, siten että tiedon merkitys ei muutu [Mykkänen *et al*., 2003]. Viimeisen 40 vuoden aikana organisaatioihin on hankittu lukuisia erilaisia tietojärjestelmiä, eikä näiden suunnitellessa ole osattu ajatella, että muutkin järjestelmät tulisivat myöhemmin uudelleenkäyttämään

samaa tietoa. Näin ollen tietojärjestelmissä käytetyt käsitteet syntyvät tietyn sovellusalueen tarpeista. Tämä voi aiheuttaa myöhemmin ongelmia yhteentoimivuuden kannalta, koska eri organisaatoissa voi olla erilaiset tietosisällöt ja käsitteet. Lisäksi tiedon käsittelyyn liityvää tietovoi olla ainoastaan tiedonkäsittelijöiden muisteissa, koska niitä ei ole ymmärretty kirjata mihinkään tekniseen dokumenttiin. [Moisio *et al*., 2005]

Tietointegraation tarkoituksena onkin jakaa kaikille yhteinen tietomalli ja tehdä käsitteet, sekä tieto jokaiselle ymmärettäväksi. Tämä tarkoittaa myös ihmisiin ja oraganisaatioihin iskostunutta tietoa, joka pitää saada sellaiseen mutooon, että tietokoneet voivat ymmärtää sitä [Moisio *et al*., 2005]. Semaattinen yhteentoimivuus on muodostunut yhä tärkemmäksi tekijäksi, sillä nykyaikaisten tietojärjestelmien koko on kasvanut ja organisaatioiden toimintaympäristömallit ovat verkostuneimpia kuin ennen.

3.4. Integraation sosiaaliset näkökulmat

Useasti tietojärjestelmien integrointi ajatellaan vain teknisenä toimenpiteenä ilman, että huomioidaan myös sosiaalisia vaikutuksia. Kuitenkin käytettävien järjestelmien muuttuessa myös henkilökuntaa pitää opastaa ja kouluttaa, koska näiden työprosessit ovat saattaneet muuttua integraation myötä tai uutta teknologiaa on otettu käyttöön. Pitää ottaa myös huomioon, miten käyttäjille ilmoitetaan tulevista uudistuksista ja miten palautetta otetaan vastaan. Bygstad ja muut [2008] esittävät neljä erilaista vaihtoehtoa integraation sosiaalisesta näkökulmasta.

Big Bang -teoriassa sekä käyttäjät että teknologia integroidaan kerralla samanaikaisesti. Tämä tarkoittaa, että käyttäjät pitää kouluttaa ja järjestelmä siirtää tuotantoon ennen kuin järjestelmän käyttöönotetaan. Menettelyn hyvänä puolena on se, että projektia voidaan hallita tehokkaasti. Huonoina puolina Big Bang -teoriassa on, että käyttäjien ja tekniikan viime hetken sisäänajo altista suuremmille riskeille. Koska organisaatio on jatkuvassa muutoksessa, on alussa vaikea tehdä kauaskantoisia päätöksiä.

Sidosryhmien integraatiossa käyttäjät ajetaan muutokseen hiljalleen koulutusten ja työryhmien myötä. Näin ollen käyttäjiltä saadaan palautetta pidemmältä aikajaksolta ja järjestelmää voidaan ohjata sen mukaisesti. Huonona puolena sidosryhmä-integraatiossa on se, että järjestelmästä voi tulla monimutkainen, koska palautteen antajia on runsaasti. Ei voida myöskään taata, että käyttäjät ovat täysin sitoutuneita koko projektin keston ajan. Käyttöönotossa saattaa myös ilmetä teknisiä ongelmia, koska tekniikka tuodaan mukaan vasta projektin loppuvaiheessa.

Tekniikan integroinnissa pyritään siihen, että järjestelmien komponentit pyritään ajamaan uudistukseen mahdollisemman aikaisessa vaiheessa. Tämän hyödyt näkyvät järjestelmän teknisenä vakautena, koska kehitystä voidaan tehdä koko projektin ajan. Huonoina puolina tästä voidaan todeta, että käyttäjän näkökulma on saattanut projektin aikana unohtua ja sisäänotto voi synnyttää kitkaa uuden järjestelmän ja käyttäjän välillä. Suunnittelusta voi kehittyä myös liian monimutkainen, sillä tekniset yksityiskohdat ovat heti alusta lähtien mukana.

Viimeisenä mallina Bygstad ja muut [2008] esittävät sosioteknisen integrointimallin. Tämän ideana on, että tekniikan ja loppukäyttäjän välillä on dynaaminen suhde, jonka avulla integraation osapuolen muutokset saadaan paremmin huomioon. Loppukäyttäjät saadaan myös kokeilemaan lopullista tuotetta kehitysvaiheessa, jonka ansioista saadaan palautetta ja käyttäjät pystyvät muuttamaan toimintaprosesseja tuotantovaiheeseen. Tämän mallin heikkoutena on vaikea prosessin ohjaus, koska käyttäjät ja teknologia ovat niin läheisesti vuorovaikutuksessa keskenään. Näin ollen ei voida olla täysin varmoja, mihin suuntaan kehitys etenee ja tuodaanko integraatiolla toivottuja prosessihyötyjä

4. Case - Lapin keskussairaala

Olen tutustunut Lapin keskussairaalaan ja siellä toimivaan perinnejärjestelmään. Lapin keskussairaala toimii Rovaniemellä ja siellä työskentelee noin 1500 ihmistä. Vuosittain keskussairaalassa hoidetaan muun muassa 130 000 avohoitokäyntiä ja hoitopäiviä kertyy 97 000. Hoitotyön tukemisessa tietojärjestelmillä on keskeinen rooli.

Tarkastelun keskiössä on ollut VAX-pohjainen perinnejärjestelmä, jonka ensimmäinen versio on otettu käyttöön vuonna 1987. Järjestelmässä on kaksi palvelinta, VAX 7000-720 ja VAX 7000-610, jotka ovat klusterissa keskenään. Palvelimissa on tällä hetkellä käytössä OpenVMS 6.2 -käyttöjärjestelmä. Aikanaan VAX-järjestelmää on käytetty suoraan niin sanottujen ”tyhmien päätteiden” kautta, mutta nykyään palvelimen kanssa kommunikointi tapahtuu tietokoneille asennetuilla ssh-ohjelmalla. Järjestelmän käyttöliittymä on nykystandardien mukaan vanhanaikainen, sillä käyttöliittymä on pelkästään merkkipohjainen.

VAX-järjestelmän on ollut yksi Lapin keskussairaalan historian keskeisimmistä tietojärjestelmistä. Järjestelmän tärkeimmät palvelut ovat olleet potilashallinta kokonaisuudessaan ja röntgenin sekä leikkaussalin toiminnanohjausjärjestelmät. Järjestelmältä on myös tilattu potilaisiin liittyviä arkistopapereita ja tehty osastojen ruokatilauksia. Mainitut palvelut ovat vieläkin osittain toiminnassa, mutta jotain VAX:n korvaavia järjestelmiä on otettu jo käyttöön. Aikanaan VAX:lla on toiminut myös muun muassa sairaalan sähköposti, laboratoriojärjestelmä ja henkilökunnan tekstinkäsittely- ja taulukkolaskentaohjelmistot, mutta nämä on myöhemmin korvattu erillisillä järjestelmillä ja ohjelmistoilla.

4.1. Järjestelmän nykytila

Tällä hetkellä Lapin keskussairaalassa on käytössä erilliset potilashallinnan ja leikkaussalin toiminnanohjausjärjestelmät. Tämän lisäksi sairaalassa on käytössä nykyaikainen integrointialusta, jonka tarkoituksena on saavuttaa usean eri sukupolven laitteista ja ohjelmistoista koostuvan IT-ympäristön integrointi. Integrointialusta vastaa viestinvälityksen osalta aikaisemmin esitellyn koordinaattorin periaatteita mahdollistaen kuitenkin muutakin kuin pelkän viestinvälityksen. Tämä muutos on tapahtunut osittain tämän työn havainnointivaiheen aikana.

Oman perehtymiseni alkuvaiheessa VAX-järjestelmä ja siinä toimivat palvelut oli integroitu viiteen keskussairaalassa käytettävään keskeiseen tietojärjestelmään. Nämä olivat

potilaskertomus-, potilashallinta-, laboratorio-, röntgen- ja apuvälinepalveluiden järjestelmät. Integrointi VAX:n ja uusien järjestelmien välillä oli toteutettu pääsääntöisesti siltaperiaatteella, jossa erinäiset ohjelmistokoodit muunsivat tiedot vastaanottavan järjestelmän ymmärtämään muotoon.

VAX:n integrointi laboratorion ja apuvälinepalveluiden hallintajärjestelmään oli, ja on yhä, toteutettu siten, että VAX-järjestelmään on asetettu eräajoja, jotka muodostavat yksinkertaisen tekstitiedoston halutuista tietueista. Luotu tietue lähetetään ftp-tiedostonsiirtomenetelmällä kohdejärjestelmään, jossa vastaanottava palvelin muuntaa tekstitiedoston SQL-tietokannan ymmärtämään muotoon. Potilaskertomusjärjestelmän ja VAX:n viestinvälitys oli aiemmin toteutettu hyödyntäen socket-liittymää, jonka avulla potilaskertomusjärjestelmä suoritti kyselyn VAX:n tietokannasta ja vastaus lähetettiin samaa socket-liittymää hyödyntäen. Vuonna 2011 käyttöönotettu potilashallintajärjestelmä on yhdistetty VAX:iin käyttäen uutta integrointialustaa, joka muuntaa tulevat viestit käyttäen H7-standardia. Kandidaatintyön alkuvaiheessa potilashallintajärjestelmää ei ollut mahdollista integroida kaikilta osin potilaskertomus- tai laboratoriojärjestelmien kanssa. Näin ollen tieto piti ensimmäisenä siirtää potilashallintajärjestelmästä integraatioalustaan ja sen kautta VAX:lle. VAX:lla suoritettavien ajojen kautta tieto oli muiden järjestelmien saatavilla. Sama prosessi toimi myös toiseen suuntaan.

Tällä hetkellä käytössä olevaan potilaskertomusjärjestelmään on saatu vaadittava päivitys, jotta tietoa voidaan lähettää suoraan potilashallintajärjestelmään integrointialustan kautta. Potilashallintajärjestelmän ja laboratoriojärjestelmän tiedonvälitys tapahtuu yhä joiltain osin VAX:a hyödyntäen.

Näin ollen, vaikka uusia tietojärjestelmiä on tullut, VAX:n merkitys ei ole kaikilta osin loppunut. Järjestelmää on tarvittu joltain osain uusien tietojärjestelmien välisen integroinnin toteutukseen ja sen tietokantaa on käytetty hyväksi tiedon varastointiin. Uuteen potilashallintajärjestelmään ei ole myöskään saatu vaadittavaa tukea kuntalaskutukselle, jonka vuoksi tämä suoritetaan yhä VAX:n kautta. Vaikka käytössä olevat palvelut ovat vähentyneet, VAX:sta itsestään on ruvennut muodostumaan Hub-and-Spoke -mallin mukainen keskitetty piste. Hub-and-Spoke -malli on yhteneväinen koordinaattoriin perustuvan integroinnin kanssa, jossa tietty järjestelmä tai sovellus hallitsee järjestelmien välistä integrointia. Lapin keskussairaalan tilanteessa VAX on ollut koordinaattorin asemassa suhteessa uusien järjestelmien keskinäiseen integraatioon.

Pelkästään VAX-järjestelmän varassa toimii tällä hetkellä potilasasiakirjojen ja sairaalaruoan tilaus, radiologian toiminnanohjausjärjestelmä ja väestörekisterikeskuksen tietojen lähetys ja haku. Asiakirjojen ja ruoan tilaus on integroitu potilashallintajärjestelmän kanssa siten, että hoitohenkilökunnan kirjatessa sairaalaan tulevia potilaita tapahtuu myös ruokien ja asiakirjojen tilaus. Radiologiset tutkimukset tilataan joko suoraan tai potilashallintajärjestelmää hyväksikäyttäen VAX:lta, joka on yhteydessä kuvantamislaitteisiin radiologian kuvantamisjärjestelmän kautta. Näiden lisäksi Väestörekisterikeskuksesta saadaan keskussairaalaan kerran kuukaudessa päivittyvät tiedot, joita tällä hetkellä ei voida ottaa

vastaan muulla kuin VAX-järjestelmällä. Nämä tiedot jaetaan VAX:n kautta muihin integroituihin järjestelmiin. Sairaalassa päivittyneet väestörekisteritiedot päivitetään väestörekisterikeskukselle ainoastaan VAX:n rakennetun yhteyden avulla.

Lapin keskussairaala on aikanaan tarjonnut tuottamiaan tietoteknisiä palveluita myös sairaanhoitopiirin alueella oleville kunnille. Tälläkin hetkellä yksi Lapin kunnista käyttää keskussairaalan VAX-palvelimilla olevaa leikkaussalin toiminnanohjausjärjestelmää, jonka vuoksi palvelua täytyy ylläpitää.

4.2. Järjestelmän tulevaisuus

Markkinoilla on tällä hetkellä valmiina korvaavia tietojärjestelmiä jokaiseen VAX:n tarjoamaan palveluun, mutta näitä ei ole todettu kovin laadukkaiksi tai kustannustehokkaiksi kalliiden käyttöönotto- ja ylläpitokustannusten vuoksi. VAX-pohjainen järjestelmä on taas hyvin varmatoiminen ja halpa ylläpitää, jonka vuoksi sairaalalla ei ole varsinaista kiirettä päästä järjestelemästä eroon. Ainoastaan palvelimien rikkoutuessaan voi varaosien heikko saatavuus tuottaa ongelmia.

VAX-järjestelmän keskeiseen rooliin on kuitenkin tulossa muutoksia, sillä muihin tietojärjestelmiin tulevat päivitykset ja käyttöönotettu integraatioalusta vähentävät VAX:n viestinvälityksen tarvetta. Esimerkiksi potilashallintajärjestelmä tullaan päivittämään niin, että väestörekisteritietojen käsittely tulee jatkossa onnistumaan. Tällä hetkellä käytössä oleva kuntalaskutusjärjestelmä tullaan uusimaan ja ratkaisu uudesta toteutustavasta tehdään vuoden 2013 aikana. Radiologian toiminnanohjausjärjestelmä, ruokatilausjärjestelmä ja potilasasiakirjojen tilausjärjestelmä korvataan uusilla järjestelmillä ja näiden käyttöönotto pitäisi tapahtua parin vuoden sisällä.

Lapin sairaanhoitopiirin tietohallinnon visiona onkin, että VAX-pohjainen perinnejärjestelmä voitaisiin lopettaa vuoden 2015 mennessä. Tätä ennen pitäisi kuitenkin saada vähintään kolme tai neljä uutta tietojärjestelmää ja saattaa ne tuotantovaiheeseen. Tämän lisäksi uudet ja jäljelle jäävät vanhat järjestelmät pitäisi integroida soveltuvalla tavalla keskenään, joten työtä varmasti riittää, kun otetaan huomioon julkisen sektorin hankintaperiaatteet. Tällä hetkellä markkinoilla olevat palvelut eivät kuitenkaan ole kaikilta osin sopivia ja tilanne alueellisen sairaanhoidon kannalta epäkypsä, jotta voitaisiin tehdä lopullisia päätöksiä VAX-järjestelmän suhteen. Tämän vuoksi kehitys halutaan pitää hallittuna, kuitenkin samalla tarkastella mahdollisia uusia tietojärjestelmiä VAX:n korvaajaksi. Vuoteen 2015 mennessä pitää myös sairaanhoitopiirin kuntien käyttämät palvelut korvata uusilla, jotta perinnejärjestelmä voidaan lopullisesti poistaa käytöstä.

5. Yhteenveto

Tässä työssä on tarkasteltu erilaisia kirjallisuudessa esiteltyjä määrityksiä keskeisille tietojärjestelmien termeille ja integraatioteorioille, kun kohdealueena on tietojärjestelmien integrointi. Kirjallisuutta on tuettu kohde-esimerkin välityksellä, jossa tarkasteltiin suoma-

laisen keskussairaalan perinnejärjestelmän historiaa, nykytilaa ja mahdollista tulevaisuutta.

Tutkielman aikana on nostettu esille erilaisia integraation hyötyjä ja ne ovat monella osa-alueella kiistattomat. Tietojärjestelmien integrointi ja integroitavuus on myös jollain nykyaikaisen liiketoiminnan osa-alueella välttämätön edellytys, jos tarkoituksena on pärjätä alati tiukentuvassa kilpailussa. Terveydenhuollossa tietojärjestelmien integraatio mahdollistaa hoidon tehostumisen, mikä johtaa puolestaan verovarojen säästöön. Potilaille tämä näkyy laadukkaampana hoitona ja potilasturvallisuutta parantavana tekijänä.

Kuten tutkielman aikana on todettu, tietojärjestelmien integrointi ei kuitenkaan ole erityisen helppo tehtävä, varsinkaan silloin, kun integroitavana on vaikeaselkoinen ja tekniikaltaan jo mahdollisesti vanhentunut perinnejärjestelmä. Integraation kompleksisuutta kuvastaa myös se, että alan tutkijoilla tai kirjallisuudella ei tunnu olevan täysin yhteneväistä termistöä koskien integraatioprosessia. Tämän lisäksi sairaaloissa käytettävien tietojärjestelmien integraatiota vaikeuttaa montaa muuta sektoria tiukempi lainsäädäntö ja turvallisuusvaatimukset. Vanha tietojärjestelmä rajoittaa myös käytettävissä olevia integraatiotapoja, jonka seurauksena prosessista voi tulla todella kallis. Jossain tapauksissa kokonaan uuden järjestelmän luominen onkin järkevämpää, kuin vanhan perinnejärjestelmän ylläpito ja jatkokehitys.

Suomessa pääsääntöisesti verovaroilla kustannettavassa terveydenhuollossa ei myöskään jatkuvasti voida korvata kriittisiä järjestelmiä uusilla. Tämä johtaa siihen, että keskeisten järjestelmien käyttöikä on pitkä ja integraatiota muihin järjestelmiin tapahtuu paljon. Aikanaan tätä ei ole osattu ottaa huomioon tarpeeksi kauaskantoisesti, eikä näin ollen sairaalatietojärjestelmien osalta ole vaadittu tarpeeksi laadukasta standardisointia. Tämä puolestaan on johtanut tiedon pirstoutumiseen ja suureen tietojärjestelmien määrään. Viimeisten vuosien aikana tähän on kuitenkin saatu muutos, sillä tietojärjestelmien integrointia on edesauttanut kehittyneet viestintästandardit, integrointialustat ja palvelukeskeisen arkkitehtuurin yleistyminen.

Viiteluettelo

[Bennet, 1995] Keith Bennet, Legacy systems: Coping with success. *IEEE Software* 12, 1 (1995), 19–23.

[Beyer *et al*., 2004] Mario Beyer, Klaus A. Kuhn, Christian Meiler, Stefan Jablonski and Richard Lenz, Towards a flexible, process-oriented IT architecture for an integrated healthcare network. In: *Proc. of ACM Symposium on Applied Computing* (2004), 264-271.

[Bygstad *et al*., 2008] Bendik Bygstad, Peter Axel Nielsen and Bjørn Erik Munkvold, Four integration patterns: a socio-technical approach to integration in IS development projects. *Information Systems Journal* **20** 1 (2010), 53-80.

[Haux, 2006] Reinhold Haux, Health information systems — past, present, future. *International Journal of Medical Informatics* **75** (2006), 268-281.

[Kuhn and Giusse, 2001] K. A. Kuhn and D. A. Giuse, From hospital information systems to health information systems. *Methods of Information in Medicine* **40** (2001), 275-287.

[Lehman, 1980] M.M. Lehman, On understanding laws, evolution, and conservation in the large-program life cycle. *Journal of Systems and Software* **1** (1980), 213–221

[Moisio *et al*., 2005] Riku Moisio et al., *Sanastot siltana saumattomalle tiedonvaihdolle*. Edita Prima Oy, Helsinki, 2005

[Mykkänen *et al*., 2003] Juha Mykkänen, Jari Porasmaa and Juha Rannanheimo, Mikko Korpela, A process for specifying integration for multi-tier applications in healthcare. *International Journal of Medical Informatics* **70** (2003), 172-182.

[Nunamaker and Briggs, 2011] Jay F. Nunamaker Jr. and Robert O. Briggs, Toward a broader vision for information systems. *ACM Transactions on Management Information Systems* **2,** 4 (2011), 1-12.

[Saboniiha *et al*., 2012] Nazanin Sabooniha, Danny Toohey and Kevin Lee, An evaluation of hospital information systems integration approaches. In: *Proc. of International Conference on Advances in Computing, Communications and Informatics* (2012), 498-504.

[Sommerville, 2010] Ian Sommerville, *Software Engineering*, 9th ed. Addison- Wesley, 2010.

[Tähtinen, 2005] Sami Tähtinen, *Järjestelmäintegraatio - Tarve, vaihtoehdot, toteutus.* Talentum, Helsinki, 2005.

Verkkokaupan ROI:n laskentamalli

Ari Varpenius

Tiivistelmä.
Tämä tutkielma käsittelee verkkokauppainvestoinnin ROI:n (Return On Investment) laskemista pohjautuen kirjallisuuskatsaukseen. Käyn aihetta läpi pohtien verkkokauppaprojektin ja ylläpitovaiheen kustannusten muodostumista sekä verkkokaupan tuomia hyötyjä yritykselle. Esittelen myös yhden kirjallisuudessa esitetyn verkkokaupan ROI:n laskentamallin.

Avainsanat ja -sanonnat: verkkokauppa, ROI, verkkokauppainvestoinnin takaisinmaksu, verkkokauppainvestoinnin kannattavuus.
CR-luokat: K.4.4, J.1

1. Johdanto

Tilastokeskuksen tuoreen tutkimuksen mukaan kaksi kolmasosaa suomalaisista on ostanut verkkokaupasta tuotteita tai palveluita viimeisen vuoden aikana. Aktiivisimman ryhmän verkkokauppojen käyttäjistä muodostavat 24–35-vuotiaat kuluttajat. Vielä 64–75-vuotiaistakin peräti 13 prosenttia oli ostanut hyödykkeitä tai palveluja verkkokaupasta kuluneiden kolmen kuukauden aikana [Tilastokeskus, 2012]. Vuonna 2011 suomalaiset kuluttajat ostivat verkkokaupan kautta yli 10 miljardin edestä tuotteita ja palveluita [TNS Gallup, 2012]. Näiden lukujen valossa voidaankin todeta, että verkkokaupan yleistyminen, käytön laajentuminen ja verkkokaupan kautta ostettavien tuotteiden ja palvelujen arvot ovat jo niin merkittävät, että yritysten kannattaa analysoida verkkokaupan mahdollisuudet vakavasti.

Yrityksen näkökulmasta verkkokauppa voi olla joko yrityksen ainoa myyntikanava tai yksi kanava muiden myyntikanavien, kuten kivijalkakaupan, postimyynnin tai puhelinmyynnin rinnalla. Verkkokauppa voi tuoda pienille, keskisuurille ja suurille yrityksille uusia liiketoimintamahdollisuuksia ja sen avulla yritys voi saavuttaa huomattavasti aiempaa suuremman asiakasmäärän. Yritykset voivat myös palvella asiakkaitaan entistä laajemmin, nopeammin ja tehokkaammin verkkokaupan avulla. Esimerkiksi pienelle erikoistavaraa myyvälle yritykselle verkkokauppa antaa hyvän välineen tavoittaa merkittävän määrän potentiaalisia asiakkaita maan rajojen ulkopuoleltakin.

Verkkokauppainvestointia suunniteltaessa ja investoinnin päätösprosessissa yrityksillä on käytössään vaihteleva kirjo erilaisia menetelmiä verkkokaup-

painvestoinnin kannattavuuden mittaamiseksi. Moni yritys on jättänyt perustamatta verkkokaupan, vaikka sen perustaminen olisi investoinnin kannattavuuden ja takaisinmaksun näkökulmasta ollutkin järkevää. Toisaalta myös moni yritys on perustanut verkkokaupan, vaikka se ei olisikaan investointina vaikuttanut järkevältä.

ROI (Return On Investment) on yleinen laskentamalli, jolla yritys voi arvioida investoinnin kannattavuutta ja sitä, missä ajassa investointi voisi maksaa itsensä takaisin [Amberg and Hirschmeier, 2004]. Verkkokaupan ROI:n laskeminen ei ole niin yksinkertaista kuin esimerkiksi toiminnanohjausjärjestelmien osalta. Verkkokaupan ROI:ssa on huomioitava normaalien euromääräisten mittarien lisäksi myös muita näkökulmia, jotka eivät ole helposti mitattavissa. Verkkokaupan ROI:n laskentaan ei ole muodostunut yleisesti käytössä olevaa mallia, enkä löytänyt aiheeseen liittyen juurikaan tutkimuksia.

Itse olen perehtynyt aiheeseen erityisesti oman työkokemukseni kautta. Olen työskennellyt viimeiset 11 vuotta sähköisen liiketoiminnan parissa ja olen ollut mukana myymässä, valvomassa, johtamassa, konsultoimassa ja toteuttamassa verkkokauppaprojekteja. Olen työssäni huomannut, että paitsi yrityksillä, myös verkkokauppatoimittajilla pitäisi olla käytössään työkalu, jolla verkkokauppainvestoinnin kannattavuutta voisi laskea.

Tämän kirjallisuuskatsauksen tavoitteena on esitellä verkkokauppainvestoinnin yritykselle tuomia hyötyjä sekä tuoda esiin niitä kustannuksia, joita verkkokauppahankinnasta yritykselle tulee. Lisäksi esittelen yhden verkkokauppahankintoihin sovelletun ROI:n laskentamallin, jonka Amberg ja Hirschmeier [2004] ovat tehneet.

Kuvaan tutkielmassani ensin verkkokaupan roolia yrityksen liiketoiminnassa. Tämän jälkeen käyn läpi verkkokauppaprojekteihin liittyviä suoria ja epäsuoria kustannuksia sekä ylläpitovaiheen kustannustekijöitä. Kustannusten jälkeen käsittelen verkkokaupan avulla saatavia euromääräisiä ja muita hyötyjä. Kustannusten ja hyötyjen kuvaamisen jälkeen keskityn verkkokaupan kannattavuuteen ja takaisinmaksuun liittyviin asioihin sekä kuvaan Ambergin ja Hirschmeierin [2004] tekemän verkkokaupan ROI:n laskentamallin. Päätän tutkielmani yhteenvetoon ja pohdintaan.

2. Verkkokauppa osana yrityksen liiketoimintaa

Tässä luvussa käydään läpi verkkokauppaa yleisesti ja pohditaan verkkokaupan toiminnallisuuksia.

2.1. Verkkokaupasta yleisesti

Verkkokauppa on yrityksen ja kuluttajan välistä (B2C) tai kahden yrityksen välistä (B2B) kaupankäyntiä. Verkkokauppa on pääsääntöisesti toimialariippumatonta ja sitä onkin nykypäivänä kaikilla toimialoilla [Hübner, 2008]. Suomalaisilla yrityksillä verkkokaupparatkaisuja on ollut jo 1990-luvun puolivälistä lähtien, joten ne ovat jo vakiinnuttaneet asemansa tilauskanavana sekä yritysten että asiakkaiden näkökulmasta.

Verkkokaupasta käytetään Suomessa myös termejä sähköisen kaupankäynnin ratkaisu, sähköisen liiketoiminnan ratkaisu, verkkopalvelu, sähköisen asioinnin ratkaisu ja tilausjärjestelmä. Suomalaisten yritysten keskuudessa verkkokauppa-termi mielletään usein pelkästään B2C-kaupankäynniksi ja olen kuullutkin usean B2B-yrityspäättäjän toteavan, etteivät he ole hakemassa verkkokauppaa, vaan ”ihan vain” tilausjärjestelmää. Mikä sitten on verkkokauppa? Verkkokauppa on yksinkertaisuudessaan internetissä toimiva sivusto, jonka kautta asiakkaat voivat tilata yrityksen tuotteita tai palveluita [Lu, 2003]. Verkkokaupassa on tilaamisen lisäksi muitakin toiminnallisuuksia, joista Bergendahl [2004] on nostanut neljä verkkokaupan päätoiminnallisuuksiksi:

- tuoteominaisuuksien ja -tietojen näyttäminen
- tuotteiden saatavuustietojen näyttäminen
- palvelun ehdot
- tuotteiden tilaaminen ja maksaminen.

Yllä lueteltu lista pätee hyvin sekä B2C- että B2B-verkkokauppaan. Kokemukseni mukaan suurimmassa osassa suomalaisista verkkokaupoista tuoteominaisuudet ja tiedot ovat hyvin esillä. Yrityksen taustajärjestelmiin integroiduissa B2B-verkkokaupoissa saattavat tuotetiedot puuttuakin. Tuotetietojen puuttumiset johtuvat näissä tapauksissa siitä, ettei niitä ole koskaan yrityksen toiminnanohjausjärjestelmään laitettu. Tuotteiden maksaminen on Suomessa keskittynyt pitkälti verkkopankki- ja luottokorttimaksamiseen. Mielestäni suurin syy tähän on se, että suomalaiset pankit ja Luottokunta tarjoavat hyvin standardoidun, helpon ja turvallisen tavan maksurajapintojen tekemiseen ja maksamiseen.

2.2. Verkkokauppa asiakastyötä tukemassa

Verkkokauppa on toki muutakin kuin pelkästään tilaamista internetistä ja tuoteluettelon näyttämistä asiakkaille. Verkkokauppa mahdollistaa myös uudenlaisia vuorovaikutustapoja asiakkaiden ja yritysten välillä [Hübner, 2008]. Suomessa on vielä tänä päivänäkin yrityksiä, jotka ottavat tilaukset vastaan faksilla, puhelimella tai sähköpostilla. Nämä yritykset hoitavat kommunikoinnin asiakkaiden kanssa käymällä asiakkaiden luona kertomassa uusista tuot-

teistaan ja neuvottelemassa hinnoista sekä maksu- ja toimitusehdoista. Tällaisten yritysten asiakkaat saavat pääosin siis palvelua arkisin virka-aikaan tai sellaisina aikoina, kun asiakkaan yhteyshenkilö yrityksen puolelta on saatavilla. Verkkokaupan avulla yritykset voivat tarjota asiakkailleen ympärivuorokautisen tavan saada tietoa tuotevalikoimista, tuotteiden hinnoista, saatavuuksista ja tilaushistoriatiedoista. Asiakkaiden nähdessä verkkokaupan kautta yllä lueteltuja asioita vapautuu myyjien aikaa varsinaiseen myyntityöhön [Bergendahl, 2004].

Verkkokauppa tukee varsinaista myyntityötä edeltävää vaihetta (pre-sales), itse verkkokauppamyyntiä (online sales) ja myös ostamisen jälkeistä aikaa (after sales) [Lu, 2003]. Varsinaista myyntityötä edeltävässä vaiheessa asiakas etsii verkkokaupan kautta tietoa kiinnostavista tuotteista tai palveluista. Ostamisen jälkeisessä vaiheessa asiakas pystyy verkkokaupan kautta näkemään tilaushistoriatiedot ja mahdollisesti tekemään reklamaation tuotteeseen tai palveluun liittyen [Ramanathan, 2010]. Kokemukseni mukaan B2C-verkkokaupan asiakas ei useinkaan tee ostostaan siitä verkkokaupasta, josta hän tietoa hakee, vaan löydettyään kiinnostavan tuotteen alkaa tehdä hintavertailua eri yritysten kesken ostaen tuotteen sieltä, mistä sen halvimmalla saa.

2.3. Yritykset ovat verkossa eri lähtökohdista

Yritykset, joilla on verkkokauppa, voidaan jakaa neljään ryhmään [Whelan and McGrath, 2002]:

1. yritykset, jotka ovat vain staattisesti verkossa
2. yritykset, jotka verkkokaupan avulla täydentävät perinteistä liiketoimintaansa
3. yritykset, jotka verkkokaupan avulla muuttavat liiketoimintaprosessejaan ja parantavat sitä kautta tuottavuutta.
4. yritykset, joilla kaikki liiketoiminta on keskittynyt verkkokaupan ympärille.

Monella yrityksellä on siis verkkokauppa, mutta ne eivät panosta sen kehittämiseen ja markkinointiin. Yrityksillä on mahdollisuus saada verkkokaupan avulla liiketoimintaansa uutta nostetta, ja ne voivat parantaa liiketoimintaprosessejaankin sen avulla. On myös useita tapauksia, joissa verkkokauppa on luonut yrityksen kivijalkakaupalle liiketoimintahyötyjä [Bergendahl, 2004].

Verkkokaupan ympärillä on ollut sen alkuajoilta lähtien paljon hypeä [Whelan and McGrath, 2002]. Tällä tarkoitetaan sitä, että yritys panostaa verkkokauppaan vain siitä syystä, että se on pinnalla oleva asia. 1990-luvun lopun suurin hype-vaihe aiheutti Suomessakin sen, että moni verkkokauppaa harjoittava yritys joutui purkamaan verkkokauppaliiketoiminnan kannattamattomana tämän vuosituhannen alussa. Oma kokemukseni on se, että yritykset olivat

2000-luvun alkupuolella kriittisiä verkkokaupasta saatavien hyötyjen suhteen, mutta viimeisten vuosien aikana on huomattu se, että järkevällä tavalla verkkokauppaprojektiin lähtemällä verkkokaupasta on mahdollista saada liiketoiminnallisia hyötyjä.

Yritysten välinen kaupankäynti on suurin osa koko sähköistä kaupankäyntiä sekä euromääräisesti että aktiivisten verkkokauppojen lukumäärän mukaan laskettuna. Kuluttajakaupankäynti on suosituinta matkustuksessa ja vähittäiskaupassa [Hübner, 2008]. Tilanne näiltä osin elää koko ajan, mutta edelleen B2B-puolella on mielestäni helpompi saada menestynyt verkkokauppa, kuin B2C-puolella. B2B-puolella asiakkaat ovat jo valmiita tekemään tilauksia verkkokaupasta seuraavista syistä [Bergendahl, 2004]:

- vähentääkseen aikaa ja kustannuksia tuoteinformaation löytämiseen
- tehdäkseen hintavertailuja
- asiakkaat ovat kokeneet tuotteiden hintojen olevan verkkokaupassa alhaisemmat, kuin vanhojen myyntikanavien kautta tilatessa
- logistiikkakulut ovat pienentyneet ja toimitusajat lyhentyneet
- tuotteiden ja palveluiden laatuun liittyvien valitusten käsittelyajat ovat lyhentyneet.

Tämän luvun perusteella voidaan mielestäni sanoa, että nykyään verkkokauppa on vakiinnuttanut asemansa osana yritysten liiketoimintaa ja sen on varteenotettava myyntikanava perinteisten kanavien rinnalla.

3. Verkkokauppainvestoinnin kulut

Verkkokauppaprojektin kustannukset voivat olla muutamista tuhansista euroista useisiin kymmeniin miljooniin euroihin. Kustannukset riippuvat valitusta ratkaisusta, siihen tehtävistä mukautuksista sekä siitä, integroidaanko verkkokauppaa yrityksen taustajärjestelmiin [Whelan and McGrath, 2002]. Tässä luvussa käyn läpi verkkokauppaprojektin liittyviä suoria ja epäsuoria kustannuksia. Lisäksi tarkastelen ylläpitovaiheeseen liittyviä kustannuksia.

3.1. Verkkokauppaprojektin suorat kustannukset

Kokemukseni mukaan verkkokauppaprojektin suorat kustannukset voidaan jakaa seuraaviin osa-alueisiin:

- määrittely
- suunnittelu
- toteutus
- testaus
- asennus
- käyttöönotto

- katselmoinnit
- projektinhallinta
- IT-infrastruktuuri.

Yllä oleva jako on mielestäni kattava ja se kuvaa samalla myös niitä tehtäviä, joita verkkokauppaprojektissa on. Yllä lueteltujen osa-alueiden lisäksi suoria kustannuksia voi vielä tulla hankittavan verkkokaupparatkaisun lisenssistä ja sen tarvitsemista sovelluspalvelin- ja tietokantapalvelinohjelmistoista.

Verkkokaupan kustannuksiin vaikuttavat verkkokauppaan rakennettavat ja mukautettavat toiminnallisuudet. Mitä enemmän verkkokauppaan rakennetaan ja muokataan toiminnallisuuksia, sitä suuremmaksi tulevat verkkokauppaprojektin kustannukset. Verkkokaupan toiminnallisuuksilla on kuitenkin vaikutuksia asiakkaiden tyytyväisyyteen [Lu, 2003]. Kuten jo aiemmin mainitsin kohdassa 2.1, Bergendahlin [2004] mukaan verkkokaupassa on neljä päätoiminnallisuutta. Näiden toiminnallisuuksien avulla yritys pystyy näyttämään valikoimassa olevat tuotteet asiakkaille, näyttämään tuotteiden saatavuustiedot, kertomaan asiakkaalle palveluun kohdistuvat ehdot ja asiakas pystyy tekemään verkkokaupan kautta tilauksen ja maksamaan tuotteet. Nämä ovat kuitenkin vain verkkokaupan päätoiminnallisuudet, eivätkä ne välttämättä riitä menestyvään verkkokauppaliiketoimintaan. Zhuangin ja Ledererin [2004] mukaan verkkokaupassa on seitsemän ominaisuutta, joilla on vaikutuksia sivuston kokonaisvaltaiseen hyvyyteen:

- Vuorovaikutteisuus. Verkkokauppa mahdollistaa nopean tavan reagoida yrityksen tarjouksiin ja antaa mahdollisuuden yritykselle saavuttaa asiakkaita yli maa- ja maanosarajojen.
- Julkaisutyökalut. Julkaisutyökalujen avulla yritykset pystyvät informoimaan asiakkaita ja luomaan yrityksestä luotettavan kuvan.
- Yhteisöllisyystyökalut. Yhteisöllisyystyökalujen avulla yritykset pystyvät sitouttamaan asiakkaita ja saavat asiakkaat palaamaan takaisin sivuilleen.
- Tuoteluettelotyökalut. Asiakkaat pystyvät selaamaan tuoteluetteloa ja vertaamaan tuotteita. Tuoteluettelotyökalujen avulla yritykset pystyvät tekemään ristiinmyyntiä (cross-sell) ja lisämyyntiä (up-sell). Ristiinmyynti tarkoittaa asiakkaan ostaman tuotteen täydentävien tuotteiden myyntiä. Lisämyynti tarkoittaa kalliimman, mutta alkuperäiseen tuotteeseen liittyvän tuotteen myyntiä.
- Transaktiotyökalut eli ostoskori- ja tilausprosessitoiminnallisuudet. Helppo tilausprosessi on avain myymiseen. Verkkokauppaostajat ovat kärsimättömämpiä kuin kuluttajat keskimäärin. Myös turvallisuus on hyvän tilausprosessin elinehto.

- Palvelimen suorituskyky. Verkkokauppaostajien kärsimättömyyden takia palvelimen suorituskyky tulee olla hyvä.
- Käyttöliittymä. Käyttöliittymän ulkoasu antaa ensivaikutelman asiakkaalle. Rikas tuoteinformaatio ja helppo navigointi rohkaisevat käyttäjää tekemään ostoksia verkkokaupassa. Huono käyttöliittymä puolestaan hylkii ensimmäisen kerran sivustolle tulleita. Käyttöliittymän personointi on tärkeää.

Zhuangin ja Ledererin [2004] lista verkkokaupan hyvyyteen vaikuttavista toiminnallisuuksista perusteluineen on mielestäni hyvä. Siitä kuitenkin puuttuu yksi merkittävä kokonaisuus: integraatiot. Yrityksen taustajärjestelmiin integroitu verkkokauppa antaa asiakkaille vielä luotettavamman kuvan verkkokaupasta. Tämä johtuu siitä, että integraatioiden avulla asiakkaalle pystytään näyttämään reaaliaikaiset saatavuustiedot, asiakaskohtaiset hintatiedot, tilauksen statustiedot ja varmasti ajantasaiset tuotetiedot. Ilman integraatioitakin verkkokauppa toki toimii, mutta verkkokaupassa oleva tieto ei ole välttämättä ajantasaista ja se myös vaatii enemmän manuaalista työtä verkkokaupan ylläpitäjältä. Integraatioita tehdään tyypillisesti yrityksen toiminnanohjaus-, CRM- ja MDM-järjestelmiin sekä mediapankkiin. Lisäksi toimialakohtaiset vaatimukset vaikuttavat integrointeihin, kuten kassajärjestelmään integroituminen vähittäiskaupan toimialalla.

Bergendahl [2004] väittää, että jos yrityksellä on olemassa jo asiakaskäyttöön jokin julkaisu- tai muu internetissä toimiva järjestelmä, on siihen kustannustehokkaampaa tehdä verkkokauppatoiminnallisuudet kuin rakentaa täysin uusi järjestelmä. Mielestäni tämä ei pidä aina paikkansa. Kokemukseni mukaan kaikki julkaisu-, verkkokauppa- ja sähköisen asioinnin järjestelmät eivät täysin tue hyvän verkkokaupparatkaisun vaatimuksia ja niiden mukauttaminen yrityksen haluamaan suuntaan voi olla jopa kalliimpaa kuin uuden järjestelmän käyttöönottaminen. Mahdollisia verkkokaupparatkaisuja on Suomen markkinoilla todella paljon. Osa ratkaisuista on globaalien ICT-yritysten tekemiä, osa avoimen lähdekoodin ratkaisuja ja osa ratkaisuista on suomalaisten yritysten itse toteuttamia ja lisensoimia. Näiden eri ratkaisuiden vertaaminen keskenään on mielestäni haastavaa, koska kaikissa ratkaisuissa löytyy Bergendahlin [2004] esille nostamat verkkokaupan päätoiminnallisuudet. Näiden ratkaisuvaihtoehtojen erot löytyvät mielestäni hinnoittelusta, mukautettavuudesta ja niiden kyvykkyyksistä integraatioiden toteuttamiseen.

3.2. Verkkokauppainvestoinnin epäsuorat kustannukset

Epäsuoria kustannuksia voi aiheutua koulutuksesta, tehtävien uudelleenjärjestelyistä, rakenteellisista muutoksista, auditoinneista ja järjestelmässä olevista virheistä [Whelan and McGrath, 2002]. Kaikki yllä luetellut epäsuorat kustan-

nukset ovat kokemukseni mukaan tyypillisiä verkkokauppaprojekteissa. Yritysten on kuitenkin vaikea ottaa näitä kustannuksia huomioon verkkokauppaprojektia budjetoidessa ja investointilaskelmia tehdessä. Whelan ja McGrath [2002] ovat myös huomanneet, että kustannuksia tulee myös olemassa olevan datan muuntamisessa asiakkaiden ymmärrettävään muotoon. Tämä on mielestäni hyvin tyypillinen kustannus silloin, kun tehdään yrityksen toiminnanohjausjärjestelmään integroitua verkkokauppaa. Kokemukseni mukaan yritysten toiminnanohjausjärjestelmässä olevat tuotetiedot on syötetty siitä näkökulmasta, että ne ovat käytössä vain yrityksen sisäiselle henkilöstölle. Verkkokauppaprojektiin lähdettäessä yrityksen pitää siis aloittaa tuotetietojen (tuotteen perustiedot, tuotekuvaukset ja tuoteryhmittelyt) läpikäynti ja muokkaus. Mikäli yrityksellä on kymmeniä tuhansia eri tuotteita toiminnanohjausjärjestelmässä, niin työmäärä tuotetietojen muokkaamiseen voi olla erittäin suuri. Tämä toki antaa myös hyvän tilaisuuden puhdistaa toiminnanohjausjärjestelmän tietoja ja sitä kautta tuo yritykselle hyötyjä.

Mielestäni epäsuoriksi kustannuksiksi pitää laskea myös mahdolliset virheet, jotka on tehty määrittely- ja suunnitteluvaiheessa. Määrittely- ja suunnitteluvaiheet ovat äärimmäisen tärkeitä verkkokauppaprojektin osia. Mikäli yritys pyrkii vähentämään kustannuksia näistä vaiheesta, johtaa se usein ongelmiin tuotantovaiheessa. Ongelmat ovat sellaisia, että huomataankin jonkin yrityksen prosesseille tärkeän toiminnallisuuden puuttuvan tai toimivan puutteellisesti tai verkkokauppaprojektissa ei ole huomioitu yrityksen toiminnoille merkityksellisiä asioita riittävällä tarkkuudella.

Moni yritys päivittää liiketoimintaprosessejaan verkkokauppaprojektin yhteydessä, mikä myös lisää verkkokauppaprojektin epäsuoria kustannuksia. Liiketoimintaprosessien päivittäminen ei tuota tuloksia, jollei verkkokauppaa ja taustajärjestelmiä ole integroitu keskenään [Zhuang and Lederer, 2004]. Tämäkin siis korostaa integrointien tärkeyttä verkkokauppaprojekteissa.

3.3. Ylläpitovaiheen kustannukset

Ylläpitovaiheen kustannukset alkavat siitä vaiheesta, kun verkkokauppa on valmistunut. Ylläpitokustannukset voivat olla vuosittain sadoista euroista useisiin miljooniin euroihin [Whelan and McGrath, 2002]. Oman kokemukseni mukaan suomalaisten yritysten osalta kustannukset vaihtelevat muutamista tuhansista euroista satoihin tuhansiin euroihin vuositasolla.

Verkkokaupan ylläpitokustannukset muodostuvat seuraavista osa-alueista [Bergendahl, 2004]:

- Kehityskustannukset. Kehityskustannusten suuruuteen vaikuttavat volyymien kasvaminen, verkkokauppaan tehtävien muutostöiden luku-

määrä, sähköisen vuorovaikutuksen lisäämisen kasvattaminen ja tuotetietojen hallintaan liittyvät muutokset.

- Myynti- ja markkinointikustannukset. Markkinointikustannusten määrä on täysin tapauskohtaista. Markkinoilla jo valmiiksi olevat yritykset eivät joudu tekemään massiivisia muutoksia markkinointiin verkkokaupan takia.
- Operaatiokustannukset. Päivittäiset tehtävät verkkokaupan ylläpitämiseksi esimerkiksi tuotteiden, saatavuustiedon ja hinnan osalta. Operaatiokustannuksiin kuuluvat myös logistiikkakustannukset.
- Maksu- ja talouskustannukset. Maksu- ja talouskustannuksiin vaikuttava luottokustannukset ja maksut maksujenvälittäjille.

Täysin verkkokauppaan keskittyvä uusi yritys joutuu investoimaan markkinointiin merkittäviä määriä, jotta saa asiakkaita tilaamaan tuotteita verkkokaupasta [Bergendahl, 2004]. Olen ollut mukana muutamissa verkkokauppaprojekteissa, joissa yritys ei ole verkkokaupan lanseeraamisen jälkeen pyrkinyt aktiivisesti markkinointiviestinnällä ja kampanjoimalla ohjaamaan asiakkaita verkkokauppaan. Tällaiset verkkokaupat eivät tule menestymään kilpailluilla markkinoilla. B2C-kaupankäynnissä kuluttajien aktiivinen ohjaaminen verkkoon voi tapahtua kampanjoimalla verkkokauppaa joko yhdessä normaalin markkinointiviestinnän kanssa tai erikseen verkkokauppaan liittyvänä kampanjana. B2B-kaupankäynnissä moni suomalainen yritys on valjastanut oman myyntihenkilöstön opastamaan asiakkaitaan kädestä pitäen tilausten tekemiseen verkkokaupasta.

4. Verkkokauppainvestoinnista saatavat hyödyt

Tässä luvussa tarkastelen yrityksen saamia hyötyjä verkkokauppainvestoinnista. Yritykset saavat verkkokaupasta taloudellisten hyötyjen lisäksi myös muita hyötyä, kuten asiakas- ja henkilöstötyytyväisyyteen liittyviä hyötyjä. Yleisesti ottaen verkkokauppa auttaa yrityksiä kulujen karsimisessa ja asiakkaiden kanssa kommunikoinnissa [Lu, 2003].

4.1. Verkkokaupasta saatavat taloudelliset hyödyt

Verkkokaupan avulla yrityksen on mahdollisuus kasvattaa liikevaihtoaan myynnin lisäämisen kautta. Myynnin lisääminen verkkokaupan kautta voi tapahtua ristiinmyynnistä olemassa oleville asiakkaille ja uusasiakashankinnan kautta [Bergendahl, 2004]. Olen itse ollut mukana B2B-verkkokauppaprojektissa, jossa yritys tavoitteli pelkästään nykyasiakkaille myynnin kasvua, sillä yrityksen näkökulmasta heidän asiakaskattavuus oli jo maksimissaan. Verkkokaupan lanseerauksen myötä yritys sai tavoitteen mukaisesti myynnin kasvua

nykyisistä asiakkaista , mutta he saivat myös uusia asiakkaita vain siitä syystä, ettei kilpailijoilla ollut verkkokauppaa käytössä. Kyseisen yrityksen saamien tilausten keskimääräinen koko kasvoi noin 15 %, mikä on mielestäni merkittävä kasvu pelkästään verkkokaupan aikaansaamana.

Yrityksen on myös mahdollista saada kustannussäästöjä verkkokaupan avulla. B2B-puolella saadaan kustannussäästöjä, kun puhelintilausten sijaan voidaan tilaukset ottaa vastaan sähköisesti [Bergendahl, 2004]. Kokemukseni mukaan B2B-puolella on vielä nykyäänkin erittäin paljon yrityksiä, jotka ottavat tilaukset vastaan asiakkailta perinteisillä tavoilla (puhelin, faksi ja sähköposti) ja kirjaavat tilaukset manuaalisesti yrityksen toiminnanohjausjärjestelmään. Olen jopa törmännyt sellaiseen suomalaiseen keskisuureen yritykseen, jossa kaksi henkilöä keskittyi ainoastaan kirjaamaan asiakkailta tulleita tilauksia toiminnanohjausjärjestelmään. Jos tällaisessa tilanteessa verkkokauppaan saadaan siirrettyä edes puolet yritykselle tulevista tilauksista, tarkoittaa se yritykselle merkittäviä kustannussäästöjä.

Verkkokaupan avulla on myös mahdollista laskea myynnin, jakelun, varastoseurannan, tilaamisen ja maksamisen kuluja [Bergendahl, 2004]. Näiden kulujen laskeminen tapahtuu kokemukseni mukaan prosessien automatisoitumisen myötä. Esimerkiksi myyjien ei tarvitse huolehtia niin paljon rutiinitehtävistä, jolloin myyjille jää enemmän aikaa uusien asiakkaiden etsimiseen ja olemassa olevien asiakkaiden hoitamiseen [Zhuang and Lederer, 2004].

Bergendahl [2004] on tuonut esille, että taloudellinen näkökulma tulee täytettyä, jos nykyasiakkaille suunnattuihin markkinointikuluihin tulee kustannussäästöjä ja uusista asiakkaista saadaan liikevaihtoa. Mielestäni tämä on kuitenkin liian suppea näkökulma taloudellisten tavoitteiden saavuttamiseksi. Taloudellisiin hyötytavoitteisiin pitää mielestäni sisällyttää myös yllä mainitut muut taloudelliset hyödyt, joita verkkokaupan kautta on saatavilla.

4.2. Verkkokaupasta saatavat muut hyödyt

Lun [2003] mukaan on selvää, että verkkokauppainvestoinneilla ja siitä saadulla hyödyillä on vaikutuksia asiakastyytyväisyyteen. Haasteena on, miten verkkokauppainvestoinnin ja asiakastyytyväisyyden parantumisen yhteydet tunnistetaan. Olen Lun kanssa samaa mieltä. Kaikissa niissä verkkokauppaprojekteissa, joissa olen ollut mukana, ovat yrityksen asiakkaat todenneet asiakastyytyväisyyden parantuneen verkkokaupan myötä. Tyytyväisyyden parantuminen on tullut näissä projekteissa tietojen ajantasaisuuden parantumisesta ja siitä, että asiakas on voinut ajasta ja paikasta riippumatta katsoa tuotetietoja, tuotteiden saatavuuksia, tilaushistoriatietoja ja tehdä tilauksia. Myös Bergendahl [2004] korostaa asiakastyytyväisyyden parantumista verkkokaupan avulla, ja

hän on listannut verkkokaupan hyödyiksi asiakkuuksien näkökulmasta seuraavat asiat:

- kustannustehokas tapa uusasiakashankintaan
- asiakassuhteiden tiivistäminen
- asiakasuskollisuuden kehittäminen.

Esimerkiksi lentoyhtiöt ja matkatoimistot ovat lisänneet sekä liikevaihtoa että parantaneet asiakastyytyväisyyttä siirtäessään palvelut verkkoon. Asiakastyytyväisyys on parantunut, koska asiakkaat näkevät saatavuudet verkosta ja pystyvät tekemään kustannustehokkaasti tilauksia [Bergendahl, 2004].

Verkkokauppainvestoinnilla on myös vaikutuksia yrityksen henkilöstön tyytyväisyyteen. Verkkokaupan avulla yrityksen henkilöstön työtyytyväisyys parantuu, osaaminen kasvaa ja tieto lisääntyy [Amberg and Hirschmeier, 2004]. Olen itsekin huomannut, että henkilö, jonka työtehtävänä on aiemmin ollut tilauksien syöttäminen toiminnanohjausjärjestelmään, pystyy verkkokauppaprojektin myötä laajentamaan toimenkuvaansa ja on sitä kautta tyytyväisempi työtehtäviinsä. Toimenkuvan muutos on voinut olla siirtyminen verkkokaupan ylläpitäjäksi tai muu aiempaa monipuolisempi työ. Toki olen huomannut myös, että osa henkilöistä pelkää oman työpaikkansa puolesta, kun verkkokauppa tulee tekemään heidän työnsä.

5. Verkkokaupan ROI

Tässä luvussa käsittelen sijoitetun pääoman tuottoasteen ja takaisinmaksun laskentamalleja, verkkokauppainvestoinnin päätöksentekoa ja kuvaan yhden esimerkin verkkokaupan ROI:n laskemiseen.

5.1. Sijoitetun pääoman tuottoaste ja takaisinmaksuaika

Sijoitetun pääoman tuottoaste (ROI) on yleisin tapa investoinnin kannattavuuden laskemiseen. ROI lasketaan kaavalla tulot / kulut [Amberg and Hirschmeier, 2004].

Yrityksen pitää siis arvioida kaikki investoinnista saatavat euromääräiset tulot ja arvioida investointiin menevät kulut. Sekä tulot että kulut pitää arvioida halutulla aikavälillä, joka voi olla esimerkiksi 5 vuotta [Leppiniemi ja Lounasmeri, 2000].

Sijoitetun pääoman tuottoaste voidaan laskea joko dynaamisesti tai staattisesti. Dynaaminen tapa ottaa huomioon myös rahan arvon muutokset, kuten inflaation. Staattinen tapa puolestaan laskee vain investoinnin tulojen ja kulujen suhteen [Amberg and Hirschmeier, 2004].

Investoinnin takaisinmaksuajan laskemisessa hyödynnetään samoja tietoja kuin pääoman tuottoasteen laskemisessa. Ensimmäiseksi valitaan aikaväli, jossa

takaisinmaksuaikaa arvioidaan. Takaisinmaksuaika saadaan jakamalla valitun aikavälin kustannukset keskimääräisellä vuotuisella tuotto-odotuksella [Leppiniemi ja Lounasmeri, 2000].

5.2. Verkkokauppainvestoinnin päätöksenteosta

Yritykset tekevät harvoin päätöksiä verkkokauppaprojektiin lähtemisestä pohjautuen investoinnin takaisinmaksuaikaan. Päätökset pohjautuvat yleensä ajatuksiin: ”Investoidaan, mikäli kilpailijat ovat onnistuneet” ja ”Investoidaan pysyäksemme tekniikassa mukana” [Amberg and Hirschmeier, 2004]. Nämä ovat vielä nykyäänkin usean suomalaisen yrityksen päätöskriteerit verkkokauppaprojektiin lähdettäessä. Kokemuspohjainen päätöksenteko verkkokauppaprojekteissa on kuitenkin luonnollista, koska tällä hetkellä ei ole olemassa sellaista työkalua, millä yritys pystyisi helposti laskemaan verkkokaupan kustannukset ja siitä saatavat hyödyt lyhyellä ja pitkällä aikavälillä.

Yritysten tulisi kuitenkin selvittää verkkokaupan liiketoiminnallisia hyötyjä [Lu, 2003]. Aiemmin olen käynyt läpi yrityksen saamia taloudellisia ja muita hyötyjä verkkokaupasta ja siihen liittyviä kustannuksia. Näiden osa-alueiden pohjalta yrityksen pitäisi pystyä arvioimaan omasta näkökulmastaan, onko verkkokauppainvestointi kannattavaa. Toki yksi tärkeä kysymys on myös, ovatko asiakkaat halukkaita tekemään tuotteiden ja palveluiden tilaamisen verkosta [Bergendahl, 2004]. Tilastokeskuksen [2012] teettämän tutkimuksen mukaan ainakin suomalaiset ovat nykyään valmiita siihen.

Ambergin ja Hirschmeierin [2004] mukaan moni yritys yliarvioi investoinnin kannattavuuden turvatakseen investointipäätöksiä, mikä vaarantaa laskelmien hyväksyttävyyden ja niihin luottamisen. Tämä on todennäköisesti tyypillistä siinä tilanteessa, kun yritys on jo tehnyt periaatepäätöksen verkkokauppainvestoinnista, mutta haluaa vielä kannattavuuslaskelmien avulla varmistaa investoinnin järkevyyden. Kannattavuuden laskeminen tulisi kuitenkin joka tilanteessa pohjautua mahdollisimman realistisiin näkemyksiin tulevista hyödyistä ja kustannuksista.

Verkkokauppaprojektin ajoitus on myös tärkeä. Pitkittämällä verkkokauppaprojektin päätöstä on mahdollista saada kehittyneempiä verkkokaupparatkaisuja ja saada lisää tietoa markkinoista. On kuitenkin huomattava, että pitkittämällä projektia voi myös menettää asiakkaiden rahoja kilpailijoille [Bergendahl, 2004].

5.3. Esimerkki verkkokaupan ROI:n laskemisesta

ROI itsessään ei ole oikea tapa arvioida verkkokaupan liiketoimintahyötyjä. Verkkokauppaliiketoiminnasta saatavat liiketoiminnalliset hyödyt eivät ole kaikilta osin mitattavissa euromääräisesti, sillä verkkokauppainvestoinnissa

yksi merkittävä liiketoimintahyöty on asiakastyytyväisyyden parantuminen [Lu, 2003]. Amberg ja Hirschmeier [2004] ovat kehittäneet dynaamisen mallin verkkokaupan ROI:n laskemiseen. Heidän laskentamallinsa pohjautuu tasapainotettuun tuloskorttiin (The Balanced Scorecard). Tasapainotettu tuloskortti on tapa ohjata ja arvioida verkkokauppajärjestelmää. Se käsittää neljä ulottuvuutta:

- Prosessit
  - Liiketoimintaprosessien parantaminen ja kehittäminen.
- Asiakkaat
  - Laadullisia mittareita, kuten asiakaslojaalisuus, asiakastyytyväisyys, yrityksen imago, asiakkaiden säilyminen.
- Työntekijät
  - Laadullisia mittareita, kuten henkilöstön työtyytyväisyyden parantuminen, osaamisen kasvaminen, tiedon lisääntyminen.
- Kustannukset
  - Taloudellisia tunnuslukuja, kuten liikevaihto, palvelujen kustannukset, maksut, kustannusrakenteet.

Lähtökohtana on se, että tasapainotetun tuloskortin avulla nähdään verkkokaupan investoinnin kannattavuus. Jotta kannattavuus voidaan näiden ulottuvuuksien avulla laskea, pitää laskentamallia tarkentaa seuraavasti:

- Prosessien tehokkuus (Process Efficiency)
  - Taloudellinen hyöty prosessien tehostumisesta.
- Asiakkaan elinkaaren arvo (Customer Lifetime Value)
  - Asiakkuudesta saatavat nettotuotot yhteensä. Pitää huomioida myös tulevaisuuden kehitys.
- Työntekijöiden elinkaaren arvo (Employees Lifetime Value)
  - Kuinka paljon enemmän yritys saa työntekijöistä arvoa verkkokauppaprojektin jälkeen. Laskentakaava on samanlainen kuin asiakkaiden elinkaaren arvon osalta.
- Kokonaiskustannukset (Total Cost of Ownership)
  - Kaikki verkkokauppaprojektiin ja jatkuviin palveluihin liittyvät kustannukset, kuten IT-hankinta-, ylläpito-, lisenssi-, koulutus-, konsultointikustannukset.
- Tulevaisuuden mahdollisuudet
  - Rahalliset tulevaisuuden hyödyt investoinnista.

Näiden avulla verkkokaupan ROI:n laskentakaavasta muodostuu seuraava:

$$\frac{\text{Prosessien tehokkuus} + \text{Asiakkuuden elinkaaren arvo} + \text{Työntekijöiden elinkaaren arvo} + \text{Tulevaisuuden mahdollisuudet}}{\text{Kokonaiskustannukset}}$$

Ambergin ja Hirschmeierin [2004] laskentakaava on mielestäni verkkokauppaan kattava, mutta kysymykseksi nousee, miten hyötyjen arvot oikeasti lasketaan. Miten yritys pystyy euromääräisesti arvottamaan esimerkiksi työntekijöiden elinkaaren arvon verkkokauppaprojektin jälkeen? Entä miten yritys pystyy arvioimaan tulevaisuuden mahdollisuudet realistisesti?

6. Yhteenveto ja pohdinta

6.1. Yhteenveto

Tässä tutkielmassa toin esille kirjallisuudesta löytyviä verkkokauppaprojektin suoria ja epäsuoria kustannuksia ja verkkokaupasta saatavia hyötyjä. Esittelin lisäksi Ambergin ja Hirschmeierin [2004] luoman laskentamallin verkkokaupan ROI:sta. Näiden lisäksi olen tuonut kokemuspohjaista tietoa ja näkökulmia suomalaisista B2B- ja B2C-verkkokauppaprojekteista. Kirjallisuudesta löytyi kohtuullisen paljon tietoa verkkokauppaan liittyvistä kustannuksista ja hyödyistä, mutta varsinaisia ROI-laskentamalleja en löytänyt esittelemääni mallia lukuun ottamatta.

Verkkokaupan kannattavuutta arvioitaessa on otettava huomioon euromääräisten hyötyjen lisäksi myös muut hyödyt, jotka liittyvät esimerkiksi asiakastyytyväisyyden ja henkilöstötyytyväisyyden parantumiseen. Kustannusten osalta mukaan on otettava niin ikään itse verkkokauppaprojektiin liittyvät kustannukset, mutta myös mahdollisten prosessien kehittämiseen ja ylläpitovaiheeseen liittyvät kustannukset.

En käsitellyt tutkielmassani pilvipalveluiden (cloud computing) vaikutuksia verkkokaupan kustannuksiin ja saataviin hyötyihin. Tein tämän päätöksen siitä syystä, etten koe pilvipalveluiden kustannuksista ja mahdollisista hyödyistä olevan vielä riittävästi tietoa ja kokemuksia.

6.2. Pohdinta

Amberg ja Hirschmeier [2004] arvioivat, että huolimatta strategisesta merkityksestä verkkokauppaprojekteilla on usein negatiivinen taloudellinen arvo. Itse en tätä arviota voi sellaisenaan allekirjoittaa. Olen toki nähnyt epäonnistuneita projekteja ja kannattamattomia verkkokauppoja, mutta niihin on aina ollut jokin tietty syy. Syy on voinut olla markkinoinnin puute, yrityksen henkilöstön negatiivinen suhtautuminen verkkokauppaan tai verkkokaupan sopimattomuus yrityksen prosesseihin. On kuitenkin selvää, ettei kovinkaan moni yritys pysty täysin todentamaan, onko verkkokauppa oikeasti hyödyllinen ja kannattava kanava tilausten vastaanottamiseen, vai onko se vain pakollinen paha ja ylimääräinen kustannus. Juuri näiden epävarmuustekijöiden pohjalta olisi tär-

keää, että verkkokauppainvestoinnin kannattavuuden ja takaisinmaksuajan laskemiseen olisi olemassa työkalu, jota yritykset voisivat käyttää arvioidessaan verkkokauppainvestointiin lähtemistä ja mahdollista verkkokaupan alasajamista.

Vaikka verkkokauppaprojektin investointilaskelmat näyttävätkin, että investointi olisi järkevää, on kuitenkin mahdollista, että yritys epäonnistuu verkkokauppaprojektissaan ja näin ollen yritys ei saakaan hyötyjä verkkokaupasta. Bergendahlin [2004] mukaan mikään ei voi kompensoida sitä, mikäli verkkokaupan avajaispäivänä verkkokaupassa on vielä virheitä. Olen Bergendahlin kanssa osittain samaa mieltä. B2C-verkkokaupan lanseeraus pitää sujua erittäin hyvin ja ratkaisun pitää olla hyvin testattu ja virheetön avauspäivänä. Mikäli kuluttaja yrittää tehdä ostoksia verkkokaupasta siinä onnistumatta, on kokemukseni mukaan erittäin todennäköistä, ettei kuluttaja palaa verkkokauppaan enää pitkään aikaan uudestaan. B2B-puolella tilanne on kuitenkin mielestäni hieman toinen, sillä siellä yrityksellä on myös mahdollisuus lievästi epäonnistua verkkokaupan lanseerauksessa. Yritysten kanssa työskennellessäni olen huomannut, että asiakkaita on toki haastavampi saada käyttämään aktiivisesti B2B-verkkokauppaa, mikäli asiakkaat ovat kohdanneet verkkokaupassa aiemmin ongelmia, mutta yleensä pitkän asiakassuhteen avulla asiakkaat saadaan palaamaan verkkokaupan käyttäjiksi.

Olen ollut mukana verkkokaupan myyntiprojektissa, jossa olemme yrityksen kanssa käyneet läpi verkkokaupasta saatavat taloudelliset hyödyt ja yritys on todennut, että hyödyt olisivat vuosittain useita kymmeniä tuhansia euroja. Yritys on kuitenkin päättänyt olla lähtemättä verkkokauppaprojektiin siitä syystä, että verkkokaupan myötä yrityksessä olisi muutamia henkilöitä jäänyt vaille töitä, eikä se ollut yrityksen arvojen mukaista. Verkkokaupan kannattavuuden laskeminen ja investointipäätöksen tekeminen ei siis edes silloin ole helppoa, kun taloudelliset mittarit vahvasti ohjaavat verkkokauppainvestoinnin suuntaan.

Viiteluettelo

[Amberg and Hirschmeier, 2004] Michael Amberg and Markus Hirschmeier, Dynamic ROI calculations for eCommerce systems. In: Amberg, M., Hirschmeier M., *Digital Communities in a Networked Society.* Springer, 2004, 119-130.

[Bergendahl, 2004] Goran Bergendahl, Models for investment in electronic commerce—financial perspectives with empirical evidence. *Omega* **33**, 4 (Aug. 2005), 283-376.

[Hübner, 2008] Ursula Hübner, Introduction to eBusiness. In: Hübner U., *eBusiness in Healthcare.* Springer, 2008, 3-26.

[Leppiniemi ja Lounasmeri, 2000] Jarmo Leppiniemi ja Sari Lounasmeri, *Yritysrahoitus*. Sanoma Pro, 2000.

[Lu, 2003] Jie Lu, A model for evaluating e-commerce based on cost/benefit and customer satisfaction. *Information Systems Frontiers* **5**, 3 (2003), 265-277.

[Ramanathan, 2010] Ramakrishnan Ramanathan, E-commerce success criteria: determining which criteria count most. *Electronic Commerce Research* **10**, 2 (2010), 191-208.

[Tilastokeskus, 2012] Tilastokeskus, Tieto- ja viestintätekniikan käyttö 2012. Saatavilla http://www.stat.fi/til/sutivi/2012/sutivi_2012_2012-11-07_fi.pdf. Viitattu 28.11.2012.

[TNS Gallup, 2012] TNS Gallup, Lehdistötiedote 29.2.2012, http://www.tns-gallup.fi/uutiset.php?aid=14785&k=14320. Viitattu 10.12.2012.

[Whelan and McGrath, 2002] Eoin Whelan and Fergal McGrath, A study of the-total lifecycle costs of an e-commerce investment. A research in progress. *Evaluation and Program Planning* **25**, 2 (May 2002), 191-196.

[Zhuang and Lederer, 2004] Youlong Zhuang and Albert L. Lederer, The impact of top management commitment, business process redesign, and IT planning on the business-to-consumer e-commerce site. *Electronic Commerce Research* **4**, 4 (2004), 315-333.